Computer Aided Medical Procedures
Prof. Dr. Nassir Navab

Dissertation

Regression Optimization for Camera Localization

Linda Mai Bui

Fakultät für Informatik
Technische Universität München

Technische Universität München
Fakultät für Informatik
Lehrstuhl für Informatikanwendungen in der Medizin

Regression Optimization for Camera Localization

Linda Mai Bui

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzende(r):	Prof. Dr. Rüdiger Westermann
Prüfer der Dissertation:	1. Prof. Dr. Nassir Navab
	2. Prof. Ales Leonardis, Ph.D.

Die Dissertation wurde am 13.07.2020 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 14.12.2020 angenommen.

Linda Mai Bui
Regression Optimization for Camera Localization
Dissertation, Version 1.0

Technische Universität München
Fakultät für Informatik
Lehrstuhl für Informatikanwendungen in der Medizin
Boltzmannstraße 3
85748 and Garching bei München

Abstract

One of the main concepts of computer vision is teaching computers and machines visual perception. Similar to how humans interact with their environment, as a first step such devices must localize themselves within their surroundings. For machines this often means localizing, or estimating the pose of the visual sensor, the camera, by estimating its orientation and position in reference to the scene. Therefore, camera pose estimation has become a vital part in tasks such as SLAM, autonomous driving, robotics and navigation or augmented reality.

Due to the low costs of RGB and RGB-D cameras and their availability on most devices, a plethora of research has focused on image-based localization, assuming single RGB or RGB-D images as sole input for the localization system. However, accurate and efficient localization from a single image still remains a challenging task. Therefore, in this thesis, we cover two main aspects to optimize regression models for camera localization systems.

First, we address the uncertainty of a model's predictions arising from noisy input data or the model itself. Due to the availability of geometric information, RGB-D methods are very accurate. However, they rely on feature descriptor matching which easily fails in ambiguous environments, often introduced by texture-less regions or repetitive structures. In addition current methods depend on computationally expensive camera pose refinement strategies to handle outliers in the matched correspondences. Therefore, to address these aspects, we employ deep learning methods for confidence prediction in correspondence matching as well as uncertainty estimation in direct camera pose regression methods from RGB images and estimate continuous multimodal distributions on the 6D camera pose space that are well suited to explain ambiguities arising in the scene.

Second, feature extraction is currently a vital part of most state-of-the-art methods and can be used for retrieval or correspondence matching in camera localization applications. Regardless of whether those features are hand-crafted or learned their representation strongly influences the accuracy and robustness of the underlying system. Therefore, we evaluate the influence of deeply learned feature representations on the accuracy of predicted poses. In particular, we discuss 1) a multi-task learning framework that is leveraging the task of learning object relevant features as well as direct pose regression and 2) how a learned feature representation can be used to refine the camera pose solely relying on RGB information.

In summary, the presented methods in this thesis focus on regression approaches in camera localization applications with deep learning and solely rely on image information. Extensive evaluation and analysis show the merit of such approaches for the task of camera localization

and as a result can be applied to indoor, outdoor as well as highly ambiguous environments. In this way this thesis opens the path for intelligent computer vision systems to more precisely localize themselves in arbitrary environments and thus paves the way towards fully automatic systems for a diverse set of applications.

Zusammenfassung

Eines der Hauptthemen in Computer Vision ist es dem Computer oder einer Maschine räumliche visuelle Wahrnehmung beizubringen. Ähnlich dazu wie Menschen mit ihrer Umgebung interagieren, ist ein erster Schritt hierfür sich selbst in seiner Umgebung wiederzufinden. Für Maschinen bedeutet dies häufig ihre eigene Position oder die eines visuellen Sensors, einer Kamera, in Referenz zu einer vordefinierten Umgebung zu bestimmen, indem die Orientierung und Position des Sensors definiert wird. Daher ist die Bestimmung der Position einer Kamera zu einem essentiellen Bestandteil von verschiedenen Anwendungen wie SLAM, autonomen Fahren, Robotik und Navigation oder Augmented Reality geworden.

Aufgrund des niedrigen Kostenaufwands von RGB und RGB-D Kameras und deren Kompatibilität mit den meisten Systemen, hat sich eine Vielzahl an wissenschaftlichen Arbeiten mit der Lokalisierung einer Kamera anhand von Bilddaten beschäftigt. Diese Systeme nehmen als Grundlage ein einzelnes RGB oder RGB-D Bild als Eingabe für das Lokalisierungssystem an. Allerdings stellt die genaue und gleichzeitig effiziente Ortsbestimmung einer Kamera von nur einem aufgenommenen Bild immer noch eine schwierige Aufgabe dar. Daher werden in dieser Dissertation zwei Aspekte zur Optimierung von Regressionsverfahren zur Lokalisierung einer Kamera von Bilddaten behandelt.

Zum Einen wird das Problem der Unsicherheit in die Ergebnisse eines Modells behandelt, welche entweder durch ungenaue Eingabedaten oder Fehler des Modells selbst entstehen können. Durch die Verfügbarkeit von geometrischen Informationen liefern RGB-D Methoden sehr genaue Ergebnisse. Allerdings, basieren diese darauf, ähnliche Merkmale zwischen korrespondierenden Punkten zu finden, welche in mehrdeutigen Umgebungen, die oft durch texturlose Oberflächen oder sich wiederholende Strukturen entstehen, häufig fehleranfällig sind. Zusätzlich verwenden diese Methoden rechenaufwändige Verfahren um die Position der Kamera nachträglich zu verfeinern und so den Einfluss von fehlerbehafteten Korrespondenzen zu lindern. Unter diesen Aspekten werden in dieser Dissertation Deep Learning Methoden analysiert die zum Einen die Wahrscheinlichkeit einer korrekten Punktkorrespondenz schätzen und zum Anderen ein Maß an Unsicherheit in die Schätzung von Modellen bestimmen, welche eine Kameraposition direkt von Bilddaten bestimmen. Für diesen Zweck werden kontinuierliche multimodale Wahrscheinlichkeitsverteilungen über die sechs Freiheitsgrade einer Kamera bestimmt, welche gut dafür geeignet sind die Mehrdeutigkeiten in einem Raum oder einer Szene zu beschreiben.

Zum Anderen ist die Bestimmung von aussagekräftigen Merkmalen anhand von Bilddaten ein essentieller Teil des derzeitigen Standes der Technik und wird in Anwendungen zur

Kameralokalisierung zur Suche in vorhandenen Datenbanken und von Korrespondenzpunkten verwendet. Egal ob diese Merkmale durch spezielle Algorithmen oder durch maschinelles Lernen erstellt wurden, bestimmen diese stark die Genauigkeit und Fehlertoleranz des zu Grunde liegenden Systems. Daher werden in dieser Dissertation der Einfluss von maschinell erlernten Merkmalen auf die Genauigkeit der geschätzten Kamerapositionen analysiert. Im Genauen wird 1) ein System analysiert, welches mehrere Probleme gleichzeitig lösen kann und dadurch objektspezifische Merkmale und deren Orientierung gleichzeitig erlernt und 2) analysiert wie erlernte Merkmale dafür verwendet werden können um die Ortsbestimmung einer Kamera nur mit Hilfe von RGB Informationen der Bilddaten zu verbessern.

Zusammenfassend fokussieren die in dieser Arbeit vorgestellten Methoden Regressionsverfahren unter Verwendung von Deep Learning zum Zweck der Lokalisierung einer Kamera, basierend einzig und allein auf Bilddaten. Umfangreiche Untersuchungen and Analysen derer zeigen die Vorteile dieser Verfahren zur Bestimmung der Kameraposition und können als Resultat sowohl in Räumen als auch im Freien und in mehrdeutigen Umgebungen angewendet werden. Auf diese Art und Weise öffnet diese Dissertation den Weg für intelligente Computer Vision Systeme, welche sich besser an verschiedenste Umgebungen anpassen können und legt eine Grundlage für zunehmend automatische Systeme in unterschiedlichsten Anwendungsfällen.

Acknowledgments

The last couple of years during which I have been working on this thesis have been shaped by the input of various people and i would like to take the opportunity to express my gratitude to all of them. During this time we have shared many exciting moments but also faced many challenges together and i am grateful for each of these moments.

First of all I would like to thank Prof. Rüdiger Westermann, Prof. Ales Leonardis and Prof. Nassir Navab for being on my committee. I am especially grateful to Prof. Nassir Navab for providing me with the opportunity to follow a PhD at the Chair for Computer Aided Medical Procedures and for the valuable discussions, feedback and advice he gave. During my PhD I was able to collaborate with an amazing team at Siemens Corporate Technology, which would not have been possible without the guidance of PD Dr. Slobodan Ilic, who I was able to interact with since my master thesis. I thank Dr. Claudio Laloni for providing me with this opportunity to work and gain experience at his department at Siemens. A special acknowledgment is reserved for Shadi Albarqouni, who has played a major role in setting me on this path and has helped and guided me since my master studies. I am especially grateful to Tolga Birdal who has truthfully been an inspiration to me and with whom i was lucky to have the opportunity to work with throughout my thesis.

Throughout this journey i was able to share a lot of moments and memories with the people at the chair as well as at Siemens and i am grateful to each of them and happy to have been able to interact with all of you. Especially to Anees Kazi for the fruitful discussions about all aspects of work and life and to Haowen Deng and Sergey Zakharov who have started this journey with me at Siemens. Thank you for the many happy moments we shared, but also for supporting me when things did not go as planned. To Christoph Baur, Amal Lahiani, Agnieszka Tomczak, Shun Cheng Wu, Ivan Shugurov, Gerome Vivar, Yongheng Zhao, Johanna Wald, Giorgia Pitteri, Oliver Scheel, Fatimeh Alidoost, Leslie Casas, Yanyan Li, Nikolas Brasch, Fabian Baur, David Tan, Fabian Manhardt, Azade Farshad, Roger Soberanis and numerous others. I am grateful to have shared this experience with all of you.

Finally, I would like to thank my family and my friends, especially Kim Lan, Andi, Minh, Linh, and Korbi for their constant support and for always having my back. I am especially grateful to my parents, Lydia and Thanh Dam, who have made sure that I could follow any path in life that i would have wanted.

Contents

Part I

Introduction and Fundamentals

1 Introduction

1.1 Motivation

How do humans interact with their environment? The human perception of their surroundings is mostly based upon three of our five senses, namely sight, hearing and touch. Out of these senses our visual perception provides us with 3D information of our environment as well as our role in that environment. According to David Marr "*Vision is the process of discovering from images what is present in the world, and where it is.*" ([138], p. 3). After we have obtained this information and know what surrounds us, we can infer where we are and localize ourselves in the environment. Finally based on this process we can interact with said environment.

The human perception system has long been an inspiration and foundation of research conducted in the area of computer vision. Therefore, one could describe computer vision as the task of teaching machines to learn perception based on visual information, similar to how humans perceive their environment. Early developments in this area date back to 1982 when Spoehr and Lehmkuhle [206] as well as Marr [138] compare a human's visual system to an information processing system. Over the years, tasks like instance segmentation, object classification or action recognition, that humans seemingly apply automatically and simultaneously, have become well-studied research topics. In addition, with the increasing interest in artificial intelligence and the necessary hardware improvements, machines and computers have started to become competitive to the performance of their human counterparts or, in few cases, even better. Some of the most prominent examples for current artificial intelligence systems reside within the area of board and computer games. For instance, most recently AlphaGo [44], an AI program developed at Google, was able to beat the best human player in the popular and complex Asian board game Go. Few methods, however, have been able to present similar performance when solely relying on visual information. Very recently AI systems, which are trained on 2D pixel information of the input image as well the overall aim of increasing the games output score, were able to show competitive performance to human players on a series of popular Atari games [148, 149].

However, even though the number of possible interactions in games such as Go are highly complex, specific rules define the validity of interactions and apply to solve the task at hand. In perception tasks, such as interaction with the environment, such rules can not easily be defined as the concepts of a room or an environment depend on the specific situation. While human beings are able to learn from few examples, for example to identify an object, machines currently require huge datasets to acquire similar knowledge. Therefore, in perception tasks the performance of computers has not yet been close to reaching the level of a human being. At its core for successful interaction with the environment, lies localization. Being able to tell where you are in the environment as well as where your surroundings are, is the first step

to a core number of visual computer vision tasks and builds part of the foundation for truly autonomous and automatic systems. To name a few, such applications include augmented reality, autonomous driving, simultaneous localization and mapping (SLAM) and robotics and navigation.

In this dissertation, the task of localizing oneself in the environment is defined as localizing the camera sensor. Assuming the camera as the main sensor used for localization, the task then becomes related to understanding the connection of visual information, in the form of 2D images captured by the sensor, to the 3D environment. Specifically the aim is to estimate the six degrees-of-freedom (DoF) of a camera, describing its orientation and position in reference to a given scene. With this aspect in mind, we will now outline the broad range of applications and importance of the topic for each application as well as give a general overview of current state of the art for each application.

1.2 Applications

Localizing oneself in the environment is the basis of a number of computer-assisted applications and enables merging the virtual world of a machine or a device and the real world of us humans. To highlight a few, augmented reality, simultaneous localization and mapping, robotics and autonomous driving are most related to the work focused on in this dissertation.

With the recent advances and developments in GPU performance, deep learning methods have become increasingly popular in most computer vision application. While deep learning frameworks have so far been mostly focused on in research areas, companies and industrial applications are starting to bridge the gap between research and commercial products utilizing deep learning methods. With this aspect in mind the aforementioned applications have been no exception and have become the subject of recent research utilizing deep neural networks. As this thesis mainly addresses deep learning approaches, we now highlight the general related work of these applications in the context of localization tasks as well as specifically highlight most recent deep learning methods.

1.2.1 Virtual and Augmented Reality

Virtual and Augmented Reality have become an increasingly interesting possibility for industrial applications and social interactions. The main focus of these applications is to provide a virtual environment for the user to interact with. Depending on the application this world can either be completely virtual or augmented with the real environment by placing virtual content into the real world. To enable such augmentation the user and device must be localized within the environment. For this aim, a number of techniques have been developed over the years, such as *marker-based* and *marker-free* tracking methods as well as *head mounted displays* that often use *infrared tracking*.

Marker-based Tracking

Visual cues, in the form of markers, have commonly been deployed to specify the position of the camera with respect to the scene. In this case, markers are designed in such a way, that they can easily be detected in an image and depending on the viewpoint associated to the camera view. For this aim, the design is specifically chosen to include strong visual information, often resembling barcode like structures. Therefore, tracking essentially boils down to detection of the marker, which in turn provides feedback to the system that can be used for accurate alignment and positioning of virtual objects [102].

One obvious drawback of such methods is the requirement of positioning physical markers in the scene which limits the applicability of such methods. If the environment is known in advance and markers can be positioned the camera can accurately be tracked. However, in dynamic environments or larger-scale rooms these methods quickly become infeasible.

Marker-free Tracking and Augmented Reality on Mobile Devices

On the other hand marker-free methods solely rely on the image content to track the camera and have, among others, become a popular choice for augmented reality applications on mobile devices. For instance, Klein and Murray [111] adapt a simultaneous localization and mapping approach, PTAM [110], for efficient tracking of the camera for augmented reality applications on mobile devices. Specifically the limited computational power as well as frequently occurring problems such as motion blur are addressed.

Due to their availability, large and attractive market and already established consumer access, mobile devices have been a main focus for augmented reality applications. In those applications tracking of the device itself is a crucial step for accurate alignment of the real world and virtual content. For this aim, augmented reality libraries such as ARKit [139] for iOS and ARCore [128] as a successor of Project Tango [137] have been developed by large companies such as Apple and Google. Such libraries enable the implementation of augmented reality applications on mobile devices through device motion tracking and scene processing. Inertial sensors as well as visual odometry are used to track the device, where features are matched between image frames and used to compute the relative motion of the camera. However, the accuracy of said tracking strongly influences the user experience as tracking errors or failure can lead to misplacement of virtual content in the scene. In addition the limited computational power of such devices poses an additional challenge for such applications.

Head Mounted Displays and Infrared Tracking

Successfully used in the gaming industry to provide the user with a more interactive experience using virtual reality, head mounted displays and devices such as HTC Vive [161] or Oculus Rift [119] have been developed. For such devices head tracking and inertial measurements allow the user to change the viewpoint in the virtual world. Additional sensors such as infrared sensors and accordingly tailored devices and controls further allow tracking of the user and interaction with the virtual environment in small-scale rooms. In this case, initial calibration of the system is of crucial necessity for good user experience and accurate deployment. Further two aspects are of most importance for such devices, 1) accurate position and orientation tracking of the system and 2) low latency between physical movement of the user and update of the displayed virtual environment.

In contrast mixed reality devices such as HoloLens [43] have found their way into industrial applications and have become especially promising in planning and manufacturing. The relatively light-weight device enables interaction with the virtual content while retaining the users field-of-view and freedom of physical movement. For future applications such as virtual meetings these devices have shown to be an important step towards remote work environments, such that multiple people will be able to view the same virtual content in 3D even though physically not in the same place. Collaborators can join the virtual space of a host from standard tablet or laptop devices without the need of their own wearable device [43].

In the field of robotics augmented reality is starting to become an attractive possibility for assisting remote control of robots. By providing a virtual or augmented view of the robot's environment the user obtains additional information that can help in planning and conducting the robot's movement to solve the task at hand. Especially in medical applications 3D overlays have been shown to aid during surgical interventions for human surgeons [157, 158, 201] as well as steerable robots that enable precise operations needed during surgeries [127, 167].

1.2.2 Robotic Navigation and Grasping

Further, automation with robotics has already made its way into industrial applications such as automatic packing and transportation in warehouses. Consumer robots often aiding in house-hold task such as unloading the dishwasher or lawn mowing are either already available or currently being developed [187]. Localization in this context is two-fold, 1) the robot must navigate its way from his starting location to the target and 2) once arrived the target object needs to be localized with respect to the robot to allow for successful grasping.

Reinforcement learning [101] as well as imitation learning [91, 93, 196] have shown to be promising directions for teaching robots such tasks. In this overview, however, we focus on vision-related methods that target robotic grasping applications and are more related to the topic addressed in this thesis. Many more methods that focus on path planning and robotic steering, the kinematics of the task at hand and force control [12], however, do exist.

If the location of the object is unknown, a first step to successful robotic manipulation of the object is the detection of said object and inference of its 3D position in the scene. There has been a vast amount of research addressing this topic, ranging from general object detection methods [74, 129, 175, 176], not specifically only applicable to robotics, to full pose estimation methods based on CAD models of the objects as prior knowledge [90, 104, 164, 208, 238].

Once detected, a decision how to best grasp the object has to be made. Saxena et al. [194, 195], therefore, aim to identify good grasp locations from multiple views of an object in a supervised fashion. Without any additional knowledge, such as a CAD model or 3D reconstruction, points are located with logistic regression and triangulated from the multiple views based on visual information. Once located the robot can be steered towards grasping the target and subsequent tasks. Recently deep-learning based solutions have started to emerge to predict grasp locations from 2D images [27, 72, 121].

Robotic navigation on the other hand is still highly related to the next application we discuss within the scope of this thesis, simultaneous localization and mapping. Originating in robotic navigation, it is still a highly researched topic within the robotics as well as in the computer vision community.

1.2.3 Simultaneous Localization and Mapping

Simultaneous localization and mapping, in the beginning also mentioned as *robotic mapping* or *concurrent mapping and localization* [213], describes the task of localizing the used device or robot and at the same time creating a map or 3D reconstruction of the environment while moving in it. The two tasks, in this case, are highly correlated, as accurate localization is essential for computing a precise map and at the same time good map information is needed to estimate correct camera poses. One issue of these methods, often addressed as *drift* of the camera, is the accumulation of small errors in location of the camera over time. To prevent this, *loop closure detection* is used to detect already visited locations and use that information to correct the global map and camera locations. Global localization, which we address in this dissertation, is especially important for these applications and can help to correct drift and detect loop closures. Further, global localization can aid in recovering the system in case of tracking failure, in this context commonly called *re-localization*. Occlusions, illumination changes or rapid camera motion can cause the tracking system to fail, in which case the system would have to be re-initialized and the map rebuild. Re-localization methods can prevent such a restart of the entire system by localizing the global pose of the camera such that tracking can be resumed at that specific position.

Early works on SLAM have been divided into individual steps, including landmark detection, state estimation and state as well as landmark update. These methods then commonly solved the problem at hand using Extended Kalman Filters [4, 39, 59].

More recently vision-based solutions using stereo as well as monocular [55] camera information have emerged, solely relying on image information. *Visual odometry* plays an essential role in such methods and focuses on estimating the relative motion of a camera between image frames. Over time the relative motion between frames is computed and so called key-frames chosen according to a certain criteria, e.g. a very simple solution would be within a certain rotational and translational distance from each other. The global map is then represented as a graph, where each node is represented by a pose for each key-frame and the edges correspond to the relative motion between frames. *Global pose graph optimization* [114] further reduces errors in the map. A common division for SLAM systems is the separation between *sparse* and *dense* methods.

Sparse Methods

Sparse methods for SLAM focus on identifying sparse key-points in images and matching the features extracted at these points between frames, resulting in a sparse reconstruction of the environment in the form of a point cloud.

One popular example of such a method is ORB-SLAM [151] by Mur-Artal et al., a monocular camera approach that uses ORB [179] features for tracking, mapping, localization and loop

closure, as well as the extension of the method to a stereo-system setup [152]. Another prominent example by Klein and Murray is Parallel Tracking and Mapping [110], PTAM, which introduces parallel processing for efficient SLAM, where two threads run simultaneously, one for tracking the camera and another one for optimizing the global map. *Bundle adjustment* [217], globally optimizing the camera trajectory and generated map, is for a first time applied in a real-time scenario. Due to its high computational burden, bundle adjustment was for a long time considered to not be applicable in real-time.

Dense Methods

In comparison to sparse methods dense methods aim, as the name suggests, at using pixel-level information to create a dense reconstruction of the environment and surfaces contained in it [16, 63, 108, 160, 188].

Dense Tracking and Mapping, DTAM [160] by Newcombe et al., is one of the first works in this direction and enables the reconstruction of a dense map by multi-view reconstruction and depth estimation. In comparison, LSD-SLAM [63], by Engel et al., restricts the depth estimation to certain regions to achieve fast computational times and, as a result, a semi-dense reconstruction.

Subsequent works have focused on different aspects of the SLAM pipeline. For instance, Kerl et al. [108] estimate visual odometry by optimizing over photometric as well as geometric constraints between frames using RGB-D image streams as input. A probabilistic formulation of the error is used that additionally allows for key-frame selection based on the distance to the previous key-frame with increasing entropy.

With the advent of deep learning and re-evaluation of existing methods in almost all aspects of computer vision, SLAM approaches have been no exception and have recently been extensively studied in the context of deep learning [16, 212]. CodeSLAM [16] by Bloesch et al., for example, learns a more compact but dense representation of the scene by training an auto-encoder on depth information conditioned on the RGB intensity image. A variational auto-encoder is trained to predict depth as well as pixel-wise uncertainty values. For image pairs with unknown poses and codes, geometric and photometric losses are used to optimize both codes and relative poses.

Re-Localization in SLAM

One main problem of SLAM approaches using visual odometry is the error in estimation of the camera motion that accumulates over time, resulting in drift of the camera and an overall inconsistent camera trajectory. Re-localization in the global map, however, can be used to correct the drift. For this aim, most often key-frames are used to correct drift as well as detect loop closures. Revisiting a certain location can then, in its most simple form, be detected by comparing a frame to the already stored key-frames. In addition, re-localization enables revisiting the environment as otherwise it would not be possible to use a previously built map anymore and can prevent the need for re-building the entire map in case of tracking failure. A couple of methods presented in this thesis are specifically designed for the task of re-localization from images. Therefore, we now highlight the development and current research in camera re-localization for SLAM systems.

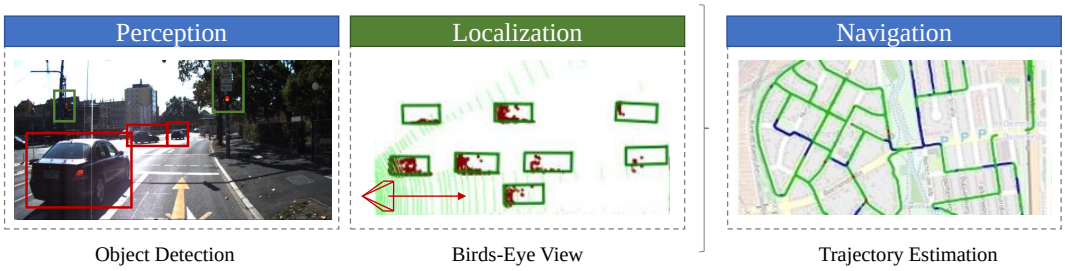


Fig. 1.1. Perception of the vehicle's environment and its localization are core requirements for autonomous control and navigation. Images are adapted from the KITTI Dataset [70].

Especially applied in key-frame based SLAM, place recognition and matching of frames to a given database has frequently been applied for re-localization, such that tracking can be resumed from the nearest neighbor key-frame [67, 69, 75]. However, the computational expenses of such methods significantly increase with the size of the map and additionally are restricted to the viewpoints contained in the database. In sparse SLAM methods, fast retrieval of corresponding matches for real-time re-localization plays an important role. Williams et al. [225] for instance propose fast matching based on decision trees and a binary encoding of features.

To alleviate the computation burden of the matching step, object-related re-localization has recently been used for indoor SLAM approaches, simultaneously building a map of objects present in the scene that are then used as landmarks [124] and matched to a query for re-localization and matching to the global map [20, 142, 181].

Recently, and most related to the work presented in this dissertation, visual localization methods have emerged addressing machine learning and deep learning techniques specifically designed to meet the requirements of image-based re-localization. For this aim, extensive research and analysis has been conducted in the last couple of years, ranging from regression forests [42, 199, 220] to deep learning methods that rely on a 3D model of the environment [21, 24] to directly regression the camera pose [105, 107].

Simultaneous localization and mapping has been a popular topic in robotic navigation and has, due to its success, found its way into several applications, one of which is autonomous driving.

1.2.4 Autonomous Driving

Self-driving cars, in general, need good navigational systems and therefore accurate localization in the environment for successful deployment. As illustrated in Figure 1.1, detection of objects in the car's surrounding such as other vehicles, pedestrians as well as traffic signs and localization of the car itself, are the core requirements for successful and safe autonomous control. The additionally large-scale nature of this application poses further challenges as the potential environment size ranges from city- to eventually world-scale. Further, in contrast to indoor applications, varying seasonal and weather conditions, such as snow, rain, fog, night and day-time pose strong illumination and structural variations. Feature-based methods that

rely on strong visual information such as geometry, edges, and lines and consistent geometric movement of such features between frames have difficulty adapting to such environments. In the context of autonomous driving, the vehicle's movement must be tracked, a process commonly addressed as *ego-motion estimation*. Further, localization of its position in the map is required, for which we highlight current works on *place recognition*.

Ego-Motion Estimation

Relative camera pose estimation as well as optical flow and depth prediction are tightly coupled tasks, restricted by geometric constraints. Therefore, a plethora of research has focused on multi-task learning frameworks in the context of autonomous systems, incorporating the three tasks into deep models and leveraging the constraints between them to steer the learning process. Focusing on ego-motion estimation, Yin and Shi [235] propose to jointly learn camera motion, depth and optical flow estimation from video sequences. Pose and depth predictions are used to obtain the rigid flow estimation and geometric as well as photometric constraints leveraged to obtain an unsupervised learning framework. Similar, Bian et al. [11] use photometric and epipolar constraints on subsequent frames to learn depth, ego-motion of the camera and optical flow simultaneously. Chen et al. [47] address how to incorporate scale-consistency into the network's ego-motion estimation to obtain global scale-consistent camera trajectories over video sequences. Geometric constraints between frames are used on the predicted depth maps to obtain an overall consistent depth prediction over the sequence. Specifically focusing on how to handle moving objects in the scene that can easily affect the models performance, Ranjan et al. [174] propose to decouple static and non-static objects.

Semantic Information

In terms of visual place recognition, a significant amount of research has focus on incorporating semantic information in the context of autonomous driving to handle the particular challenges of this application such as seasonal changes and varying weather conditions. To solve this problem, a line of research has proposed the use of semantic information, that should be invariant to the aforementioned changes. In this context, as seasonal changes should not affect semantic labels of a captured view, Schönberger et al. [198] propose to learn features capturing geometric as well as semantic information to be robust against such conditions. Based on the semantic segmentation of a query image and its depth map a 3D semantic map is build and features extracted using a variational autoencoder. The features are then used to find matches with a given database of the global 3D scene from the training set and the alignment with the best match is confirmed with the semantic labels. Toft et al. [215] extend traditional structure-based methods to include a semantic consistency score to handle erroneous 2D-3D matches. A semantically labeled structure from motion model is used to project points into the image plane and their semantic labels identified to additionally score the quality of in- and outliers. In case semantic labels are not available, the authors of [118] propose to cluster pixels according to learned features from a neural network and use the resulting classes as additional constraints similar to semantic information.

All of the aforementioned applications and methods depend on camera localization, where a strong trend towards *image-based* methods and *visual localization* is highlighted. Although many more exist, in this dissertation, we explore methods aiming at improving camera localization approaches mostly from single RGB and RGB-D images. Note that more specific

and technical related work for each framework presented in this dissertation is described in the individual chapter.

1.3 Thesis Outline

In the next chapter, we start with a brief overview of the necessary mathematical background and computer vision fundamentals this thesis is based on. In particular we cover different rotation parameterizations such as rotation matrices, quaternions and exponential maps and general rigid transformations in $SE(3)$ and $se(3)$. We outline the camera model used in this dissertation, the pinhole camera model. Further, estimation of the camera pose from 2D-3D and 3D-3D point correspondences are described and robust estimation with RANSAC briefly explained.

Part II In this part, we focus on retrieval-based methods for object recognition and pose estimation as well as large scale visual localization. We introduce a multi-task learning framework for manifold learning, that is guided by pose information to learn suitable features for the task at hand. We synthetically create object renderings and use feature learning to bridge the domain gap for successful object recognition as well as pose estimation. Further, we extend our framework to large scale visual place recognition with a geo-tagged database of RGB images. Part of this content is included in the publication

Mai Bui, Sergey Zakharov, Shadi Albarqouni, Slobodan Ilic, Nassir Navab, *'When Regression meets Manifold Learning for Object Recognition and Pose Estimation'*, Proceedings of IEEE International Conference on Robotics and Automation (ICRA).

Part III We then focus on structure-based methods, that generally rely on explicit or implicit feature matching and subsequent pose estimation from point correspondences. We introduce a general method for implicit correspondence learning from 2D-3D or 3D-3D correspondences depending on the information available. Most importantly we introduce a confidence prediction for the found correspondences with deep learning to enable robust localization in the presence of a large amount of outliers. The related publication is:

Mai Bui, Shadi Albarqouni, Slobodan Ilic, Nassir Navab, *'Scene Coordinate and Correspondence Learning for Image-Based Localization'*, Proceedings of the British Machine Vision Conference (BMVC)

Part IV In this part, we cover two methods based on direct camera pose regression.

First, we explore the possibility of RGB-based camera pose refinement in Chapter 8. Inspired by the concept of generative adversarial networks we opt for classification by a discriminator network of joint poses and image features. During inference said network is then used for iterative pose refinement pushing the regressed pose towards the manifold of the learned pose distribution. This work is covered in the publication

Mai Bui, Christoph Baur, Nassir Navab, Slobodan Ilic, Shadi Albarqouni, *'Adversarial Networks for Camera Pose Regression and Refinement'*, Workshop on Deep Learning for Visual SLAM, Proceedings of the International Conference on Computer Vision (ICCVW).

Secondly, in Chapter 9 we propose a variational learning framework suitable to capture the multimodal nature of ambiguous environments using Bingham and Gaussian Mixture Models. Such distributions are suitable to capture the pose space parameterized by quaternions and Euclidean translations and in addition provide a measure of uncertainty in the predicted poses. Further, we show how to alleviate common problems such as mode collapse that frequently arise in training such models by incorporating a multiple hypothesis training strategy. The related publication is described in

Mai Bui, Tolga Birdal, Haowen Deng, Shadi Albarqouni, Leonidas Guibas, Slobodan Ilic, Nassir Navab, *'6D Camera Relocalization in Ambiguous Scenes via Continuous Multimodal Inference'*, European Conference on Computer Vision (ECCV).

Part V We conclude this dissertation with a summary of the work presented in this thesis, possible limitations of our and current state-of-the-art methods, discussion about possible solutions and future directions and applications that have not been covered within the scope of this dissertation.

Appendix In the appendix, we provide a list of the publications covered in this dissertation as well as additional publications not discussed here. For those publications we include a short description and overview as an appendix.

2 Mathematical Background

In this chapter, we briefly describe the fundamental principles and mathematical background of this dissertation. We start with introducing commonly used rotation parameterizations that can be used to describe the orientation of a camera. We then relate the 3D world to image pixels and explain the assumed projective geometry of this thesis using rigid transformations. Lastly, we discuss commonly used algorithms to estimate a camera pose from correspondences and how a robust solution with RANSAC can be computed.

2.1 Rotation Parameterization

The orientation of a camera can be parameterized in a number of ways. Most frequently used in current state-of-the art methods are rotation matrices, quaternions or the axis-angle representation. Therefore, we will shortly summarize these parameterizations as well as highlight advantages and drawbacks that can arise when training deep neural networks for orientation regression.

Rotation matrices Rotation matrices are members of the 3D special orthogonal group, $SO(3)$, where $\mathbf{R} \in \mathbb{R}^{3\times3}$ are orthogonal matrices such that $\mathbf{R}\mathbf{R}^T = \mathbf{R}\mathbf{R}^{-1} = \mathbf{I}$ and $\det(\mathbf{R}) = 1$.

Using rotation matrices in an optimization process results in an over-representation of the otherwise three DoF of a rotation to nine DoF. Further, the orthogonality of the matrix has to be ensured during the optimization to obtain a valid solution.

Axis-Angle Equally any orientation can be represented by a rotation of an angle $\theta \in [0, 2\pi)$ around an axis $\boldsymbol{w} \in \mathbb{R}^3$, where $\theta = \cos^{-1}(\frac{\mathrm{tr}(\mathbf{R})-1}{2})$. The vector $\boldsymbol{w}$ can be derived from

$$\log(\mathbf{R}) = \frac{\theta}{2\sin\theta} \cdot (\mathbf{R} - \mathbf{R}^T). \tag{2.1}$$

Reducing the current four DoF to only three, the Lie algebra, $so(3)$, introduces a way to merge the angular and axis component into a one vector representation, $\boldsymbol{w} \in \mathbb{R}^3$. An element of $so(3)$ has the form

$$\boldsymbol{w}_1 G_1 + \boldsymbol{w}_2 G_2 + \boldsymbol{w}_3 G_3 \in so(3), \tag{2.2}$$

a linear combination of $\boldsymbol{w}$ and the skew-symmetric generator matrices

$$G_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}, G_2 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}, G_3 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{2.3}$$

Then, according to Rodrigues formula the corresponding rotation matrix of $\boldsymbol{w}$ can be computed by the exponential map:

$$\exp(\boldsymbol{w}_\times) = \exp \begin{pmatrix} 0 & -\boldsymbol{w}_3 & \boldsymbol{w}_2 \\ \boldsymbol{w}_3 & 0 & -\boldsymbol{w}_1 \\ -\boldsymbol{w}_2 & \boldsymbol{w}_1 & 0 \end{pmatrix} = \mathbf{I} + \frac{\sin\theta}{\theta}\boldsymbol{w}_\times + \frac{1-\cos\theta}{\theta^2}\boldsymbol{w}_\times^2, \tag{2.4}$$

where $\boldsymbol{w}_\times$ is the skew-symmetric matrix of $\boldsymbol{w}$ [62]. A matrix is considered skew-symmetric if $\boldsymbol{w}_\times^T = -\boldsymbol{w}_\times$

Quaternion Quaternions are vectors on the 3-sphere $\boldsymbol{S}^3$. Given a rotation of angle θ around axis $\boldsymbol{w}$, the associated quaternion can be computed as

$$\mathbf{q} = [\cos\frac{\theta}{2}, \boldsymbol{w}\sin\frac{\theta}{2}]. \tag{2.5}$$

As quaternions reside on the unit sphere, normalization is required during any optimization procedure to obtain a valid quaternion. Further, quaternions are redundant in the sense that $\mathbf{q}$ and $-\mathbf{q}$ represent the same rotation, which additionally should be addressed. Although quaternions are defined by vectors of $\mathbb{R}^4$ instead of the minimal amount of parameters required in $\mathbb{R}^3$, they have extensively been used in recent literature utilizing neural networks to estimate the orientation of an object or a camera.

When predicting samples of the aforementioned representations with neural networks, it is important to keep the properties of each representation in mind to obtain a valid solution and to not over-complicate the learning process.

Distance metrics on rotations A distance between two rotation matrices, $\mathbf{R}_1$ and $\mathbf{R}_2$, can be computed from the difference rotation matrix $\mathbf{R} = \mathbf{R}_1\mathbf{R}_2^T$ or the geodesic distance between $\mathbf{R}_1$ and $\mathbf{R}_2$:

$$d(\mathbf{R}_1, \mathbf{R}_2) = \|\log(\mathbf{R}_1\mathbf{R}_2^T)\| = \|\log(\mathbf{R})\| \tag{2.6}$$

Retrieving the angle of $\mathbf{R}$ as

$$\theta = \arccos\frac{\mathrm{tr}\mathbf{R} - 1}{2}, \tag{2.7}$$

with $\mathrm{tr}\mathbf{R} = 1 + 2\cos\theta$, provides an intuitive measure of distance between two rotation matrices.

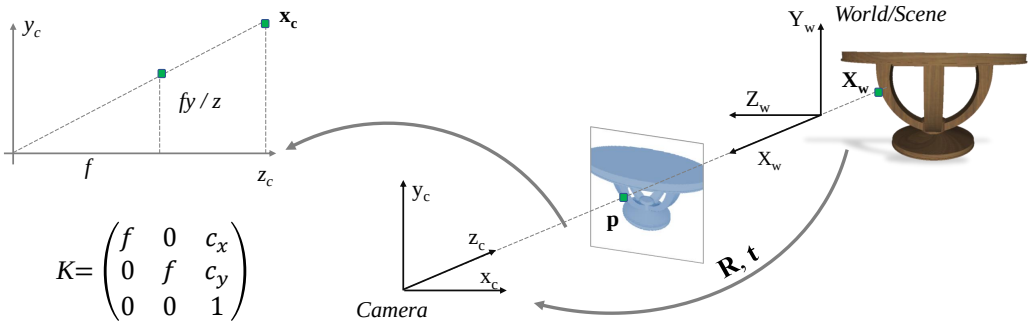


Fig. 2.1. Mapping from 3D world to a 2D image using the camera pinhole model. The extrinsic parameters, $\mathbf{R}$ and $\mathbf{t}$, of a camera describe the transformation between world and camera coordinate frame, whereas its intrinsic matrix $\boldsymbol{K}$ relates the information of 3D points to the 2D image plane.

Similar a distance between two quaternions, $\mathbf{q}_1$ and $\mathbf{q}_2$, can be computed as

$$\theta = 2\arccos(|\mathbf{q}_1\mathbf{q}_2|), \tag{2.8}$$

where $\mathbf{q}_1\mathbf{q}_2 = \cos\frac{\theta}{2}$ and is computed as the dot product between $\mathbf{q}_1$ and $\mathbf{q}_2$. Both metrics result in values of $\theta \in [0, \pi]$ [95].

2.2 Projective Geometry

The mapping between a 3D point $\mathbf{X}_w \in \mathbb{R}^3$ or later also called the *scene coordinate* and an image pixel $\mathbf{p}$ is described by the underlying camera model. One of the most commonly used models, which we will also use in this dissertation, is the *camera pinhole model*, which describes the camera by its extrinsic and intrinsic parameters. Although the model does not account for distortion, it has been widely used in computer vision applications. Now, assuming the camera pinhole model, as visualized in Figure 2.1, first $\mathbf{X}_w$ is mapped into the camera's coordinate frame using the extrinsics, or the camera pose, as $\mathbf{x}_c = \mathbf{R}\mathbf{X}_w + \mathbf{t}$, where $\mathbf{R} \in SO(3)$ represents the camera's orientation, $\mathbf{t} \in \mathbb{R}^3$ its translation and $\mathbf{x}_c \in \mathbb{R}^3$ is the 3D coordinate of point $\mathbf{X}_w$ in the camera coordinate frame. Given the camera's intrinsic parameters, the focal length f (distance between the camera center and its image plane), and its optical center $\mathbf{c} = (c_x, c_y)$ (intersection of principal axis and the image plane), the projected pixel location $\mathbf{p}$ of $\mathbf{x}_c = (x, y, z)^T$ can be computed as $\mathbf{p} = (\frac{fx}{z} + c_x, \frac{fy}{z} + c_y)^T$. $\boldsymbol{P} = \boldsymbol{K}[\mathbf{R}|\mathbf{t}]$ is called the camera matrix, with

$$\boldsymbol{K} = \begin{pmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{pmatrix}. \tag{2.9}$$

Note that in the above equation we assume the scaling factors of x- and y-direction to have equal length in pixels, f. The mapping from image pixels to 3D coordinates is commonly called *back-projection*.

2.3 Rigid Transformations

If we write $\boldsymbol{P}$ using homogeneous coordinates, we obtain a rigid transformation in the 3D space. These transformations form a group $\boldsymbol{P} \in SE(3)$ and preserve relative distances and angles among points. Using Lie Algebra any matrix $\boldsymbol{P}$ can be represented as a vector $(\boldsymbol{u}, \boldsymbol{w})^T \in se(3)$ with $\boldsymbol{u}, \boldsymbol{w} \in \mathbb{R}^3$. The exponential map defines the mapping from $se(3)$ to $SE(3)$ and can be computed as

$$\exp \begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{w} \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \boldsymbol{V}\boldsymbol{u} \\ \mathbf{0} & 1 \end{pmatrix}, \tag{2.10}$$

with

$$\mathbf{R} = \mathbf{I} + A\boldsymbol{w}_\times + B\boldsymbol{w}_\times^2 \tag{2.11}$$

and

$$\boldsymbol{V} = \mathbf{I} + B\boldsymbol{w}_\times + C\boldsymbol{w}_\times^2, \tag{2.12}$$

where $A = \frac{\sin\theta}{\theta}$, $B = \frac{1-\cos\theta}{\theta^2}$, $C = \frac{1-A}{\theta^2}$ and $\theta = \boldsymbol{w}\boldsymbol{w}^T$ [62]. $\mathbf{I}$ in this formula is the identity matrix.

Distance metric Given two rigid transformations $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$, we can define a distance metric in exponential coordinates as

$$d(\boldsymbol{P}_1, \boldsymbol{P}_2) = \|\log(\boldsymbol{P}_1\boldsymbol{P}_2^{-1})\|, \tag{2.13}$$

which can be seen as the amount of movement required to move from $\boldsymbol{P}_1$ to $\boldsymbol{P}_2$.

2.4 3D/3D and 2D/3D Camera Pose Estimation

The connection between image plane and the 3D world is defined by its geometric information and, for instance, can be leveraged to estimate the most likely camera pose connecting two point sets.

3D/3D If two sets of corresponding points $\{\mathbf{x}_c^1, \mathbf{x}_c^2, ..., \mathbf{x}_c^N\}$ and $\{\mathbf{X}_w^1, \mathbf{X}_w^2, ..., \mathbf{X}_w^N\}$ in camera and world coordinate frame are available, the associated camera pose, $\mathbf{R}$ and $\mathbf{t}$ can be recovered using *Kabsch algorithm* [100] or sometimes also called *orthogonal Procrustes alignment*. In short, the algorithm finds the optimal rotation that aligns the two sets of points by minimizing the root mean squared error (RMSE). Having obtained the optimal rotation, the translation can be recovered as a next step. Given the two 3D point sets $\{\mathbf{X}_w^i\}_{i=1}^N$ and $\{\mathbf{x}_c^i\}_{i=1}^N$ the algorithm finds an optimal solution $\mathbf{R}$ and $\mathbf{t}$ such that

$$\arg\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{N} \|\mathbf{x}_c - (\mathbf{R}\mathbf{X}_w + \mathbf{t})\|^2 \tag{2.14}$$

For this aim, the rotation matrix $\mathbf{R}$ is retrieved from the matrix $\mathbf{\Sigma}$ with covariances between the two point sets:

$$\mathbf{\Sigma} = \sum_{i=1}^{N} (\mathbf{X}_w^i - \overline{\mathbf{X}}_w)(\mathbf{x}_c^i - \overline{\mathbf{x}}_c)^T, \tag{2.15}$$

where $\overline{\mathbf{X}}_w$ and $\overline{\mathbf{x}}_c$ are the mean point of point sets $\{\mathbf{X}_w^i\}_{i=1}^N$ and $\{\mathbf{x}_c^i\}_{i=1}^N$ respectively. The rotation matrix $\mathbf{R}$ can then be retrieved from the singular value decomposition of $\mathbf{\Sigma}$ as

$$\mathbf{R}^T = \mathbf{U}\mathbf{V}^T, \tag{2.16}$$

where

$$\mathbf{\Sigma} = \mathbf{U}\mathbf{S}\mathbf{V}^T. \tag{2.17}$$

The matrix $\mathbf{S}$ is a diagonal matrix and contains the singular values of $\mathbf{\Sigma}$. The case that the matrix $\mathbf{R}^T$ results in a reflection, $\det(\mathbf{R}^T) = -1$, instead of a rotation matrix can easily be handled by changing the sign of the last column of $\mathbf{U}$. The translation vector $\mathbf{t}$ is then given by

$$\mathbf{t} = -\mathbf{R}\overline{\mathbf{X}}_w + \overline{\mathbf{x}}_c. \tag{2.18}$$

2D/3D In a lot of applications direct 3D to 3D correspondences might not be available due to the lack of depth or similar information. However, if it is possible to retrieve correspondences between the 3D world and 2D image space, the camera pose can still be approximated. This is called solving the *perspective-n-point* (PnP) problem between two sets $\{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_N\}$ and $\{\mathbf{X}_w^1, \mathbf{X}_w^2, ..., \mathbf{X}_w^N\}$. Various methods have been proposed to solve this problem, depending on if the intrinsics of the camera are known or not. If not the *Direct Linear Transform* (DLT) is one of the most popular approaches. However, in this thesis we generally assume the camera's intrinsic parameters to be known. As the camera pose has six degrees of freedom, a minimum of three corresponding points are necessary to retrieve a solution, therefore introducing the perpective-three-point problem (P3P) or in photogrammetry often called the *Resection* problem. This problem has been known for a long period of time and first solutions date back to 1841 [82]. In short to solve the P3P problem, angles between the rays from the camera center to each of the 3D points are used and equations formed that result from geometric properties of triangles between the camera center and the 3D points. Solving for these equations then retrieves the depth of each point and thus the solutions to the possible camera poses can be computed. Since at its core quadratic equations need to be solved, four possible solutions are obtained and generally a fourth point is used to solve for these ambiguities. Although this is a well- and long-known problem, much research is still conducted to develop more robust and efficient solutions such as [89, 103].

As a general solution, for $n > 3$, EPnP [122] has been proposed. EPnP expresses the point set as a weighted sum of four virtual control points and essentially reduces the problem to solving for the camera coordinates of these control points. Having obtained corresponding camera and world coordinates, as a second step the pose of the camera can be retrieved.

Handling outliers In case of perfect, error-free, data the camera pose can be retrieved using the aforementioned algorithms. However in real world applications, due to noise and errors in computation, the obtained data is almost never perfect, which can result in severe lack of performance. Therefore, strategies such as *Random Sample Consensus* (RANSAC) [66] have

become well adapted to handle outliers and compute a robust solution. An outline of the

Algorithm 1 RANSAC Pseudocode

```
Input: N data points
Output: estimated model
while True do
  randomly sample M points, the minimum number of data required;
  estimate a model;
  check which points agree with this solution based on a certain criteria;
  if good solution then
    re-estimate the solution based these points
    exit
  end if
end while
```

algorithm in its basic form is depicted in Algorithm 1. It is an iterative algorithm, that starts by computing an initial solution from the minimum amount of data required, in our case three point correspondences. Based on this initial hypothesis, the inlier set is computed over all available data points. Here, a point is considered to be an inlier if a chosen distance threshold, commonly the re-projection error, is within a user-specified threshold. The hypothesis is then recomputed based on the inlier set. Repeating this computation N times ensures finding an optimal solution with probability p, where N can be computed based on the probability of observing an inlier p_{in} as

$$1 - p = (1 - p_{in}^{M})^{N}, \tag{2.19}$$

where M is the minimal number of data points used to compute a model hypothesis. Based on its ability to handle a large amount of outliers, RANSAC is still a widely used approach in camera as well as object pose estimation or in *fundamental matrix* estimation to find the relation between correspondences in stereo images [131].

Algorithm 2 Pre-emptive RANSAC for Camera Pose Estimation

```
Input: N 2D-3D point correspondences
Output: estimated camera pose [R, t]
K times randomly sample three point correspondences;
estimate K solutions by solving the PnP;
while K > 1 do
  sample N points for each solution;
  score ← count the inliers in N for each solution;
  discard worst K/2 hypotheses;
  K ← K/2;
  refine each hypothesis on the enlarged inlier set;
end while
```

An adapted version of RANSAC, *pre-emptive RANSAC*, has been widely used in camera localization methods [41, 199, 220]. Therefore, we will shortly outline the procedure in Algorithm 2. An initial set of hypothesis is sampled using the minimum amount of required points. Then for each hypothesis a scoring function is computed, based on the inlier count, and only half of the hypotheses with the highest scores are kept and updated using their inlier set. This process is repeated until only one hypothesis remains.

2.5 Error Metrics

In this section, we define common metrics used in the remainder of this thesis across all chapters. Application or method specific metrics are further explained in each chapter separately.

Given a ground truth camera pose, consisting of a rotation, represented by a quaternion $\mathbf{q}$, and its translation $\mathbf{t}$, we compute the angular error between ground truth $\mathbf{q}$ and predicted quaternion $\hat{\mathbf{q}}$ as

$$d_q(\mathbf{q}, \hat{\mathbf{q}}) = 2 \arccos(|\mathbf{q} \circ \hat{\mathbf{q}}|). \tag{2.20}$$

For translations we use the norm of the difference between ground truth $\mathbf{t}$ and predicted translation $\hat{\mathbf{t}}$

$$d_t(\mathbf{t}, \hat{\mathbf{t}}) = \|\mathbf{t} - \hat{\mathbf{t}}\|_2 \tag{2.21}$$

to compute the error in position of the camera. We can then provide statistical measures of these errors over the entire dataset, such as the median, mean and standard deviation.

In addition, we evaluate the performance of our models with respect to the accuracy of the predicted camera poses by computing the recall of ours and the baseline models. We consider a camera pose estimate to be correct if both rotation and translation are below a pre-defined threshold:

$$\mathbf{E}[\rho((d_q(\mathbf{q}, \hat{\mathbf{q}}) < \text{th}_q) \wedge (d_t(\mathbf{t}, \hat{\mathbf{t}}) < \text{th}_t))], \tag{2.22}$$

where $\rho(\cdot)$ is 1 if its argument holds true and 0 otherwise. The thresholds are defined for rotation and translation, namely th_q and th_t, and strongly influence the measure.

In the next chapter, this definition is of particular importance for our method, as it, in addition of being used as an evaluation measure, also defines a distance function between images, which we use and incorporate in our first proposed framework for image-retrieval based on neural networks in the context of pose estimation tasks.

Part II

Pose Estimation with Image-Retrieval

3 Introduction

3.1 Motivation

For centuries image retrieval methods have been used to solve a majority of computer vision tasks such as image or object classification [112, 203] or scene recognition [204, 243]. In general these methods rely on a pre-computed database of images and associated labels, that depend on the task at hand for example classes, categories, geo-locations or camera poses. Given a query image for which the associated label is unknown a nearest neighbor search in the given database will effectively retrieve the most likely label, a process which is depicted in Figure 3.1. For successful and efficient matching, first a lower dimensional representation of the database as well as the query image is computed. As the performance of these methods naturally depends on the chosen representation, various works have focused on finding general as well as task-specific features that can be used to encode the input image. For this aim, traditional methods have relied on hand-crafted features such as *histogram of oriented gradients (HoG)* [54], that has been designed to encode specific image properties such as strong edge gradients which often relate to the shapes and surfaces contained in the environment or *scale-invariant feature transform (SIFT)* [130] and *speeded up robust features (SURF)* [8], that first detect keypoints in an image and then extract features at those locations. Resulting descriptors are designed to have certain properties, such as scale as well as rotation invariance or robustness to illumination changes such as depicted in Figure 3.2, where images of the same location have a large variation in visual appearance due to illumination conditions.

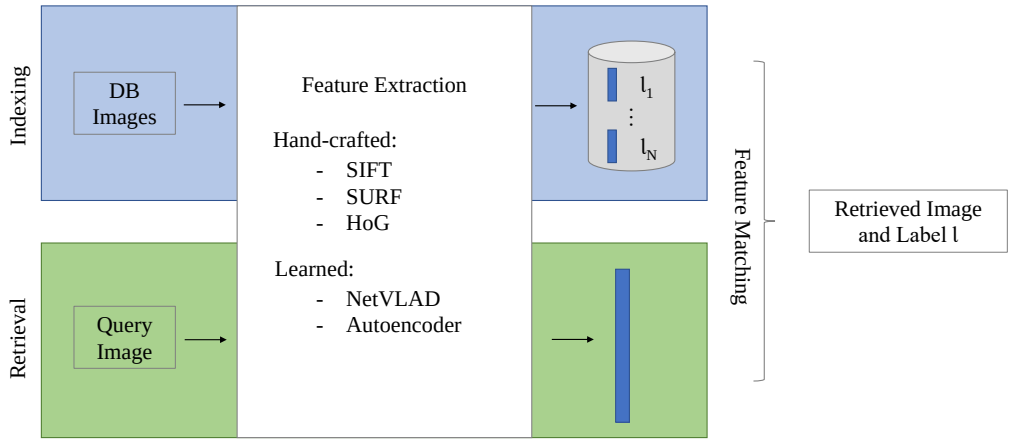


Fig. 3.1. General image retrieval pipeline. Common retrieval methods rely on feature matching between the query and a pre-computed database to retrieve the nearest neighbor and thus most likely match or label. Features originally were mostly hand-crafted to extract specific information such as edges and strong gradients. Recently the use of learned feature representations have shown to be superior in a number of applications.

Fig. 3.2. Example images of the same location under varying illumination conditions of the Tokyo 24/7 dataset [216]. In a retrieval application, extracted features should be invariant to such conditions and produce similar descriptors for successful matching.

Preferably for successful retrieval of the nearest neighbor such images should still be close to each other in the descriptor space.

Further, these often local features can be combined in a global descriptor for image retrieval using *bag-of-words* approaches. Originally a popular method used in Natural Language Processing for text retrieval, bag-of-words approaches have been used to describe the content of text documents based on the histogram of word occurrences. Similar, to describe image appearances, local features can be clustered and aggregated to obtain visual words. All visual words of an image, the bag of visual words, is then used to represent an image [168]. Matching of images is then a result of comparing the frequency of visual words appearing in two images.

However, with the advent of machine and deep learning, learned feature representations have been shown to provide powerful and rich encodings that can outperform their traditional counterparts and in addition can easily be adapted to adjust to a specific task [226]. In most cases this corresponds to adapting the loss function, such that features are learned to be close to each other if the input images or labels correspond to a predefined distance function. Assuming, for example, the task of object recognition, feature descriptors should be learned such that descriptors for the same object class appear similar whereas they should be well differentiable from features of other object classes. The distance functions used commonly directly relate to the problems at hand, such as classes, geometric or visual distance measures.

For this aim, in this chapter we present a multi-task learning framework that is specifically designed to learn a representation encapsulating what is shown in the image as well as from which viewpoint it is seen. For instance, we evaluate our method on the task of object recognition and pose estimation, where features are required to be discriminative enough to be able to differentiate between object classes regardless of their viewpoint. In addition, the viewpoint information should not be neglected as it provides important clues about the object and is required to infer the exact object's pose. Further, we evaluate the performance of our model in large-scale environments on the task of place recognition, where features have to be robust to strong illumination changes for successful recognition of the environment.

3.2 Related Work

In the context of six DoF pose estimation, the database normally consists of features extracted from RGB or depth images and associated object or camera poses. Finding the nearest neighbor of a query image according to the resulting features will in this case also retrieve the closest pose. This can be done at large scale considering geolocations as labels or in smaller scale on scenes and objects to retrieve the six DoF camera/object pose.

3.2.1 Place Recognition and Camera Pose Estimation

We first review large scale place recognition and camera pose estimation methods where the main focus is to compute a discriminative yet compact representation that can efficiently be indexed in large databases. We then move towards smaller scale object retrieval.

Place Recognition

At large-scale, i.e. city or world-scale, this problem has originally been posed as recognition of the place a query image is taken at, relying on landmarks that are often covered in a query image and associated geo-locations. Due to strong illumination changes such as day/night or seasonal changes as snow/rain, extracted descriptors have to be carefully designed to be invariant to the aforementioned effects [48, 49]. For this aim, Relja et al. [3] construct a new feature aggregation layer, inspired by the *vector of locally aggregated descriptors (VLAD)* [99], which can be included in any existing convolutional neural network and clusters local features into a global descriptor similar to bag-of-visual-words. However, instead of counting occurrences it stores the sum of residual vectors between the feature vector and its visual word (cluster center). NetVLAD [216] then shows how to incorporate VLAD descriptor into a neural network, where the visual words as well as the cluster assignment for each descriptor are introduced as learnable parameters. It shows great capabilities in aiding image retrieval tasks for place recognition and therefore has been used in several subsequent works [189, 211, 216].

On the other hand, in [29] Budvytis et al. propose the use of semantic information for effective retrieval. Here, a neural network is trained to predict semantic labels, which in turn are then used for matching with a given database by comparing the histogram of predicted labels and database images.

In contrast to RGB images, a LiDAR scan is less affected by changes in illumination or weather conditions. Therefore, numerous works propose to solve the task of place recognition from a LiDAR point cloud [2, 84, 241]. Uy and Lee propose PointNetVLAD [2], which combines PointNet [172] to learn local descriptors for each point in the cloud, that are then aggregated using a NetVLAD [3] layer and finally reduced to a lower dimensional feature representation trained on a triplet loss. In PCAN [241] the authors, Zhang and Xiao, propose to predict attention weights for the learned local features that can be used to compute a weighted aggregation. Intuitively points that correspond to dynamic objects such as pedestrians passing by should not influence the learned descriptor.

Six DoF Pose Estimation

Depending on the application, retrieving a coarse estimate of the query place or geo-location might not be accurate enough. In augmented reality applications for instance an exact pose prediction is necessary to provide a reliable user experience and accurately position virtual content into the real world environment. Therefore, Taira et al. [211] additionally propose to learn dense features using a convolutional neural network for camera pose estimation. After retrieval of nearest neighbor database images, dense features at different layers of the network are used to find 2D-3D correspondences, given that depth information or a model, for example from structure-from-motion (SfM), is available for the database images, from which the pose can be computed. Additionally, the estimated pose is verified by comparing the query image and the synthesized view obtained using the 3D model and retrieved camera pose. Similarly, Sarlin et al. propose HF-Net [189], which applies a coarse to fine strategy, first retrieving the nearest database image and then matching 2D-3D correspondences using learned features at learned keypoint locations. Knowledge distillation is used effectively to obtain an efficient model that can be employed on mobile devices. Zamir et al. [239] use a siamese network architecture to compute features based on object-centric viewpoint matches for camera pose estimation. Further, they show that the resulting model generalizes well to other tasks, including object pose estimation. Still, this method as is has not been shown to generalize to multiple objects in a cluttered environment. Balntas et al. propose RelocNet [5], a convolutional neural network aiming at learning a feature representation between image pairs that corresponds to the camera frustum overlap of the associated camera poses. In combination with a predicted relative pose, nearest neighbor camera poses can be inferred from the learned feature representation and the pose of a query image with respect to the nearest neighbor computed.

3.2.2 Object Recognition and Pose Estimation

On a smaller scale, considering objects instead of places or entire scenes, the problem becomes recognizing an object instance and/or its pose with respect to the camera. The idea, however, remains similar and can easily be applied to this very related topic.

Manifold learning

As a pioneer in this line of research Wohlhart and Lepetit [226] present a manifold learning approach for object recognition based on the triplet and pairwise losses. In their work, for each RGB-D image patch containing a specific object, triplets are formed in such a way that a query image patch is matched with a similar patch if they contain the same object, and a dissimilar patch, depicting a different object category. Features are then learned in such a way that descriptors will be close in Euclidean space if the image patches contain the same object and highly dissimilar otherwise. Zakharov et al. [236] include in-plane rotations to the above mentioned method and improve it by introducing an updated triplet loss function in which the margin term value is set to be dynamic depending on the type of the negative sample, as opposed to the former method. Sundermeyer et al. [208] propose a convolutional auto-encoder to learn representative features solely based on image appearance which can then be used to retrieve the most similar pose of a query patch. Concurrently to our work Balntas et al.[5] show the positive effect a pose-guided neural network can have on its learned feature representation by enforcing features to be correlated to the distance in pose space

of the object's pose. These methods, except for [236], however, do not include in-plane rotations.

Domain Adaption and Randomization

Obtaining a high amount of annotated images with pose labels is necessary for supervised learning methods. While realistic renderings can nowadays be created thanks to modern visualization and rendering techniques, the creation of such data remains a time-consuming and tedious task. Therefore to alleviate this problem synthetic images can be generated from predefined poses. CAD models of objects are very often available and a theoretically infinite number of rendered images with corresponding poses can easily be computed. Unfortunately, due to illumination conditions clutter and occlusion occurring in real environments, these renderings often significantly differ from real images, which can easily reduce the performance of deep learning models to be applied in real world applications, a problem commonly known as *domain shift* [52].

Several methods have therefore aimed at solving the domain shift problem. Very often a limited amount of real labels is available or can be obtained, introducing *domain adaption* and *domain randomization* methods.

Domain Adaption

Specifically in deep learning methods, it has been shown that neural networks can, to some extent, generalize to the real data when trained on synthetic data in addition to a small amount of real ones. To further improve this effect explicit manifold learning can be applied to push features from different domains, e.g. real and synthetic, to be close in the features space. This technique for instance has been shown to work well on the task of object recognition [226].

In case no real images and labels are available, generative models have been another widely used approach to handle the domain shift problem. For instance real images can be created from synthetic ones [200, 210] or vice versa [98, 208, 237] or models directly trained to map from one domain to a target one [18]. These methods in general aim at bridging the domain shift by finding common feature representations across domains [218] or domain-specific ones [19] that can then further be processed or used to acquire additional data. A drawback of such methods, however, is that they require paired samples from the source to the target domain, which might not easily be available. Therefore, a line of research has focused on domain adaption techniques that learn from unpaired examples [109, 248].

Domain Randomization

On the other hand domain randomization has been proposed to introduce variability in the training set and only rely on synthetic renderings that can still be used to generalize to real data. For this aim, a line of research [208, 236] applies heavy data augmentation, such as varying image contrast and brightness or inserting random background images, for example from the Pascal VOC dataset [64], in addition to mimicking occlusions. Such augmentations have shown to perform well on the task of object pose estimation. At medium to larger scale synthetic renderings of indoor scenes are created for robotic applications [17, 185, 214].

The method presented in this thesis utilizes both domain randomization as well as domain adaptation on the task of object pose estimation. This way, our method is able to leverage a large amount of synthetic renderings to learn a model for object recognition as well as pose estimation. In the next chapter we will therefore explain and describe our proposed method in more detail and show extensive evaluation on the task of object pose estimation. Further, we provide experiments on the task of place recognition, showing the generalization possibilities of our method to several applications.

4 Manifold Learning with Regression Optimization

In this chapter we now describe our proposed method for object recognition as well as pose estimation based on manifold learning, before showing its generalization capabilities on the task of place recognition. The following work has, in parts, been published in the following paper

'When Regression Meets Manifold Learning for Object Recognition and Pose Estimation' by Mai Bui, Sergey Zakharov, Shadi Albarqouni, Slobodan Ilic and Nassir Navab, International Conference on Robotics and Automation (ICRA), 2018, Copyright 2018 IEEE [36].

4.1 Methodology

Our methodology starts with training a convolutional neural network on a given training set $S_{train} = \{\mathbf{s}_1, \ldots, \mathbf{s}_N\} = \{(\mathbf{X}_1, C_1, \mathbf{q}_1), \ldots, (\mathbf{X}_N, C_N, \mathbf{q}_N)\}$, which consists of N samples. Each sample $\mathbf{s}$ comprises a depth image patch $\mathbf{X} \in \mathbb{R}^{n \times n}$ of dimension $n \times n$ of an object, that can be identified by its class $C \in \mathbb{N}$, together with the corresponding pose vector $\mathbf{p} = [\mathbf{q}, \mathbf{t}]$, where we focus on estimating $\mathbf{q} \in \mathbb{R}^4$. We choose to represent the pose as the orientation of the object, parameterized by quaternions. An overview of our method is depicted in Figure 4.1. Note that we leave the detection of the object to future work. However, any state-of-the-art

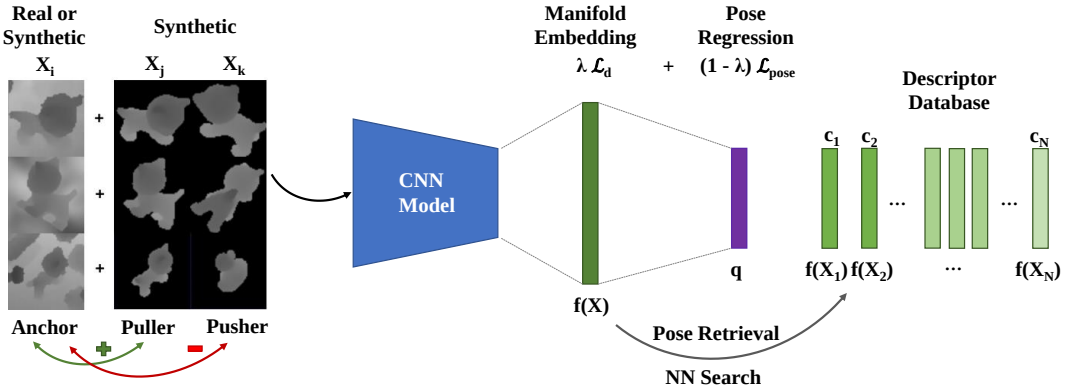


Fig. 4.1. Given an input depth image patch $\mathbf{X}_i$, we create corresponding triplets $(\mathbf{X}_i, \mathbf{X}_j, \mathbf{X}_k)$ and pairs $(\mathbf{X}_i, \mathbf{X}_j)$ to optimize our model on both manifold embedding, creating robust feature descriptors, and pose regression. Obtaining either a direct pose estimate $\mathbf{q}$ or using the resulting feature descriptor for nearest neighbor search in the descriptor database.

object detection method that provides a 2D bounding box such as [175] could be applied here and used to retrieved the translation $\mathbf{t}$.

Therefore, our objective becomes the task of modeling the mapping function $\mu_{\mathbf{\Gamma}}(\mathbf{X}) : \mathbb{R}^{n \times n} \mapsto \mathbb{R}^4$, such that for a given input patch $\mathbf{X}$ the predicted pose vector $\hat{\mathbf{q}}$ is obtained as

$$\hat{\mathbf{q}} = \mu(\mathbf{X}; \mathbf{\Gamma}), \tag{4.1}$$

where $\mathbf{\Gamma}$ are the model parameters, in our case the weights of a neural network. While the primary objective is to obtain an accurate pose estimation for any unseen data, having a well clustered feature space is of high interest as well such that our model can scale to multiple classes, in comparison to recent state-of-the-art methods that train one model per object. The resulting features can then be used to identify the objects class. For this aim, we model the problem as a multi-task learning framework, namely we train our model to simultaneously solve the two tasks, pose regression and descriptor learning. Finally, we formulate the overall objective function as

$$\mathcal{L}_{MTL}(\mathbf{\Gamma}|\mathbf{X}, \mathbf{q}) = (1 - \lambda)\mathcal{L}_{pose}(\mathbf{\Gamma}|\mathbf{X}, \mathbf{q}) + \lambda\mathcal{L}_{d}(\mathbf{\Gamma}|\mathbf{X}), \tag{4.2}$$

where λ is a regularization parameter. $\mathcal{L}_{pose}(\mathbf{\Gamma}|\mathbf{X}, \mathbf{q})$ and $\mathcal{L}_{d}(\mathbf{\Gamma}|\mathbf{X})$ are the objective functions for the pose regression and the descriptor learning task respectively that we will describe in more detail in the following.

4.1.1 Pose Regression

During training, our neural network maps a given input image patch $\mathbf{X}$ to a lower dimensional feature vector $f_{\mathbf{\Gamma}}(\mathbf{X}) \in \mathbb{R}^d$. As depicted in Figure 4.1, we obtain the feature vector as the output of the last fully connected layer before it is further utilized to regress the pose in a subsequent layer. To finally regress the pose we use the following loss function:

$$\mathcal{L}_{pose}(\mathbf{\Gamma}|\mathbf{X}, \mathbf{q}) = \|\mathbf{q} - \frac{\hat{\mathbf{q}}}{\|\hat{\mathbf{q}}\|}\|_2^2, \tag{4.3}$$

where $\|\cdot\|_2$ is the l_2-norm and $\mathbf{q}$ is the corresponding ground truth pose, a loss function commonly used in recent pose regression literature [107].

4.1.2 Descriptor Learning

To create robust feature descriptors that can be used for object recognition as well as pose estimation, object identities as well as poses should be well distinguishable in the feature space and create a compact clustering of object classes as well as a respective pose mapping within the clusters. Further, to reduce the requirement of real annotated data and to be able to mainly train our model on synthetic images, we need to map synthetic and real images to the same domain to ensure generalization of our model to real applications. Here, we make use of the triplet and pairwise loss, originally introduced by Wohlhart and Lepetit [226]. For

this aim, the loss functions encapsulate both class separation as well as domain adaptation aspects.

As depicted in Figure 4.1, our model is trained on a set of triplets $(\mathbf{s}_i, \mathbf{s}_j, \mathbf{s}_k) \in T$, where the sample $\mathbf{s}_i$, called *anchor*, corresponds to the current depth image $\mathbf{X}_i$ and $\mathbf{s}_j$ is chosen so that the image corresponds to the same object C_i viewed from a similar pose $\mathbf{q}_j$. For this aim, we call $\mathbf{s}_j$ the *puller*. However, $\mathbf{s}_k$ is chosen so that the image $\mathbf{X}_k$ corresponds either to a different object C_k or the same object C_i, but viewed under a very different pose $\mathbf{q}_k$ and is therefore called the *pusher*. The resulting loss, $\mathcal{L}_{triplets}(\mathbf{\Gamma}|\mathbf{X})$, defined over a batch of triplets is formulated as

$$\mathcal{L}_{triplets}(\mathbf{\Gamma}|\mathbf{X}) = \sum_{(\mathbf{s}_i,\mathbf{s}_j,\mathbf{s}_k)\in T} \max\left(0, 1 - \frac{||f_{\mathbf{\Gamma}}(\mathbf{X}_i) - f_{\mathbf{\Gamma}}(\mathbf{X}_k)||_2^2}{||f_{\mathbf{\Gamma}}(\mathbf{X}_i) - f_{\mathbf{\Gamma}}(\mathbf{X}_j)||_2^2 + m}\right), \tag{4.4}$$

pulling viewpoints under similar poses close together and pushing dissimilar ones or different objects further apart. As appeared in [236], m corresponds to a dynamic margin defined as:

$$m = \begin{cases} 2\arccos(|\mathbf{q}_i \cdot \mathbf{q}_j|) & \text{if } C_i = C_j, \\ \gamma & \text{else,} \end{cases} \tag{4.5}$$

where $\gamma > 2\pi$. The dynamic margin ensures that objects of different classes get pushed farther away while the margin for the same objects depends on the angular distance between the current viewpoints $\mathbf{q}_i$ and $\mathbf{q}_j$.

In addition, the pair-wise loss $\mathcal{L}_{pairs}(\mathbf{\Gamma}|\mathbf{X})$ is used to push together the sample feature descriptors of the same object under the same or very similar pose but with different backgrounds or coming from different domains, i.e. synthetic or real. The pair-wise loss is computed on pairs $(\mathbf{s}_i, \mathbf{s}_j) \in P$ and is defined as:

$$\mathcal{L}_{pairs}(\mathbf{\Gamma}|\mathbf{X}) = \sum_{(\mathbf{s}_i,\mathbf{s}_j)\in P} ||f_{\mathbf{\Gamma}}(\mathbf{X}_i) - f_{\mathbf{\Gamma}}(\mathbf{X}_j)||_2^2, \tag{4.6}$$

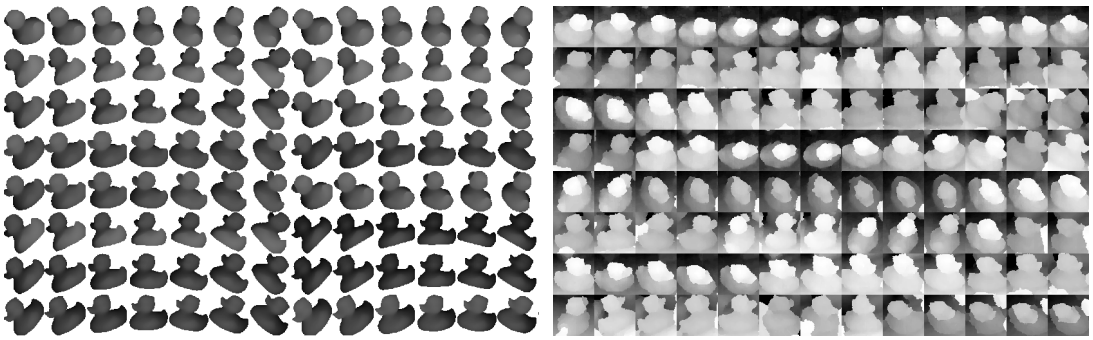
$f_{\mathbf{\Gamma}}(\mathbf{X}_i)$ being the feature descriptor extracted from the neural network for image $\mathbf{X}_i$. Overall, for descriptor learning we obtain the following loss function $\mathcal{L}_d$, where

$$\mathcal{L}_d(\mathbf{\Gamma}|\mathbf{X}) = \mathcal{L}_{triplets}(\mathbf{\Gamma}|\mathbf{X}) + \mathcal{L}_{pairs}(\mathbf{\Gamma}|\mathbf{X}), \tag{4.7}$$

consisting of pairwise $\mathcal{L}_{pairs}(\mathbf{\Gamma}|\mathbf{X})$, as well as triplet $\mathcal{L}_{triplets}(\mathbf{\Gamma}|\mathbf{X})$, loss functions.

4.2 Experimental Setup

In this section, we first describe how we create our datasets by combining simulated object renderings with background noise and real images taken from the LineMOD dataset [90]. Then we give an overview of our experimental setup before demonstrating and evaluating the results.



(a) Training Renderings **(b)** Test Patches

Fig. 4.2. Example patches for our a) synthetically rendered training and b) real test sets.

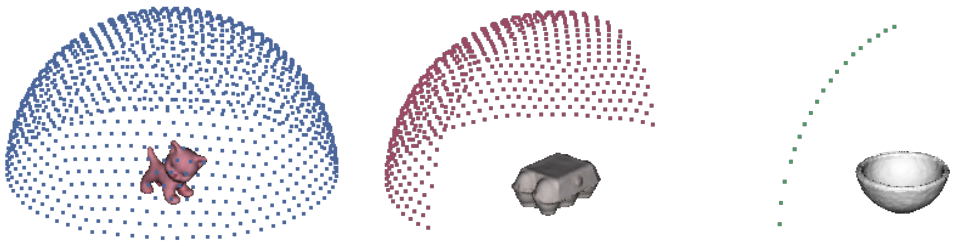
4.2.1 Dataset Generation

The LineMOD dataset is a by the state-of-the-art widely used dataset. Considering that both former closely related feature descriptor learning and pose estimation methods [226, 236] utilize the LineMOD dataset for their experiments as well, we chose this dataset to be able to evaluate our algorithm's performance in comparison to the baseline method.

The LineMOD dataset consists of fifteen distinct 3D mesh models and respective RGB-D sequences of them together with the camera poses. This data is used to create the training S_{train}, the database S_{db}, and the test set S_{test}. Each set consists of samples $\mathbf{s} = (\mathbf{X}, C, \mathbf{q})$, where $\mathbf{X}$ stands for a depth image patch, C and $\mathbf{q}$ are the corresponding object class and pose, respectively. The training set S_{train}, as its name suggests, is used exclusively for training. The test S_{test} and database set S_{db} are exclusively used in the evaluation phase, where the samples of S_{db} are used to construct a descriptor database used for the nearest neighbor search that is matched against samples from S_{test}. Examples patches for our synthetically created rendering as well as the real test patches can be found in Figure 4.2.

We now describe in more detail how our training, database and test sets are created. First, we render each of the fifteen objects from different viewpoints covering their upper hemisphere, depicted in Figure 4.3a. Here, following previous methods [226, 236], viewpoints are defined as vertices of an icosahedron centered around the object. By repeatedly subdividing each triangular face of the icosahedron additional viewpoints and a denser representation can be created. In one subdivision the midpoint of each triangle edge is chosen as a new vertex such that the new edges between the midpoints form additional faces. In our case, the training set S_{train} is sampled by recursively applying three consecutive subdivisions on the initial icosahedron structure. The resulting viewpoints can be seen in Figure 4.3a. Furthermore, following the method of [236], we add in-plane rotations at each vertex position and rotate the camera from -45 to 45 degrees using a stride of 15 degrees.

As a next step, we extract patches from the depth sequences covering the objects from both rendered and real images. The bounding box is defined by a bounding cube of size 40 cm^3 centered on the object. All the values beyond the bounding cube are clipped. Upon extraction



(a) Regular **(b)** Symmetric **(c)** Rotation-inv.

Fig. 4.3. Sampling points for different objects types: vertices represent camera positions from which the object is rendered. We choose different sampling strategies for symmetric as well as rotation invariant objects to handle ambiguous viewpoints.

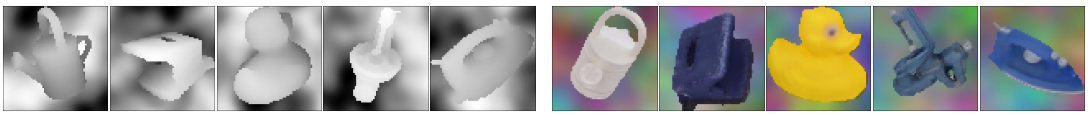
of the patches, each of them is normalized and stored together with its object class C and pose $\mathbf{q}$. As a result we obtain a single sample $\mathbf{s}$.

The training set S_{train} is then generated by combining the samples from the renderings, created as described above, and 50% of the real data, obtained from the depth sequences. This results in approximately 18% in the training set S_{train} being real data. The selection of the real samples is performed by choosing the most similar poses to the ones used for synthetically rendered samples. The remaining real samples are used to generate the test set S_{test}, ensuring variation from the training set with respect to the contained poses. The database set S_{db} is created by using only the synthetic part of the training set S_{train}.

Treating Rotation-invariant Objects

Four out of fifteen objects of the LineMOD dataset have a property of rotation-invariance and introduce an ambiguity to the generation of triplets needed for the triplet classification loss. For instance, the *bowl* object is fully rotation-invariant, whereas the *cup*, *eggbox* and *glue* object are only rotation-invariant around a single plane or, in other words, symmetric. Therefore, in comparison to the others, these four objects need to be handled with care, such that stable network training can be ensured. This stems from the fact that, in the case of rotation-invariant objects, samples representing different poses might visually look the same. This introduces ambiguous views, which can in turn result in a faulty triplet required for the triplet loss function as well as hinder the network to accurately learn the object's pose.

To solve this problem, we restrict the viewpoints rendered for those objects, such that every image patch is ensured to be unique. Sample vertices for different object types are demonstrated in Figure 4.3b and 4.3c. Since both training set S_{train}, and test set S_{test} also include real samples we omit ambiguous poses in them and only retain those that are close to the rendered samples. Note that this sampling also results in an unbalanced percentage of real images included for each object. Therefore, to consider this in our evaluation, we create datasets of five, ten and fifteen objects and only include the rotation-invariant objects when using the full LineMOD dataset of fifteen objects.



(a) Noise augmented depth patches **(b)** Noise augmented rgb patches

Fig. 4.4. Example depth and RGB renderings with augmented noise as background information.

Data Augmentation

The synthetic samples coming from the renderer have a black background, which introduces a domain gap between synthetic and real samples. Since we have a limited amount of the real data available and cannot cover all the possible poses to train our model, we augment the training samples with a background noise whenever the real sample for this pose is not available. This way, we ensure the network learns a mapping corresponding to the pose information regardless of the background information given. Augmented samples are included in the training set S_{train} during training in an online fashion, generating different noise patterns for each anchor sample on the fly. The noise type we use for our pipeline is purely synthetic and is present in both [226] and [236] showing the best performance among synthetic noise types. It is referred to as fractal noise and is based on a combination of several octaves of simplex noise first introduced in [166]. It provides a smooth non-interrupting noise pattern and is often used for landscape generation by game developers. We visualize examples of our augmented patches in Figure 4.4.

4.2.2 Implementation Details

With this training and testing setup, we extract patches of size $n = 64$. We then train our convolutional neural network, where, if not stated otherwise, we use the network architecture introduced in [226], with the single exception of our feature descriptor dimensionality being set to $d = 64$. As it is mentioned in [226] the methods performance saturates with increasing feature descriptor size. As for our multi-task learning framework we found a similar effect during our experiments, in which we experienced $d = 64$ to be a good trade-off between nearest neighbor and regression performance. During our experiments, we set $\gamma = 10.0$ for the dynamic margin. The network was trained on a Linux-based system with 64GB RAM and 8GB NVIDIA GeForce GTX 1080 graphics card. All experiments are implemented using TensorFlow [1] with Adam optimizer and an initial learning rate of $1e^{-3}$, while the batch size was set to 300.

4.2.3 Baseline Models

To analyze the performance of our model as well as the effect of multi-task learning, i.e. regression and learning robust feature descriptors together, we report the results in comparison to the baseline method [236]. For this aim, we train a model on the loss function $\mathcal{L}_d(\mathbf{\Gamma}|\mathbf{X})$ and compare the results by obtaining the nearest neighbor pose. The baseline is in the following abbreviated as *NN*. However, in comparison, the baseline method in their original work uses

RGB-D data and includes normals as additional information whereas we only rely on the depth information. Including surface normals as an additional input could give extra information and can be included in our model. We leave this investigation to future work. Nevertheless, all experiments were run using a Python implementation of their pipeline provided by the authors, so that we were able to run and compare the results using depth images only. In addition, we conduct our own baseline for a regression only model (R), trained solely to regress the object's pose, and report the results. Furthermore, to evaluate our method, we report the results obtained by our multi-task learning framework, where we report both the end-to-end regression (*Rours*), as well as the results obtained by using the resulting features for nearest neighbor retrieval (*NNours*).

4.2.4 Evaluation Metrics

To evaluate the performance of our method, we use the angular error comparing the ground truth pose $\mathbf{q}$ and the predicted one $\hat{\mathbf{q}}$ for a given image $\mathbf{X}_i$:

$$\theta(\mathbf{q}_i, \hat{\mathbf{q}}_i) = 2 \cdot arccos(|\mathbf{q}_i \cdot \hat{\mathbf{q}}_i|). \tag{4.8}$$

In our framework, the pose can either be obtained by nearest neighbor lookup or directly estimated by the neural network. Furthermore, we report the angular accuracy, where each query image is considered a true positive if its angular error is below a threshold t. For this aim, we choose threshold of $t \in [10, 20, 40]$ degrees.

Finally, to qualitatively evaluate the resulting feature descriptors, we visualize the features in a lower dimensional space for which we use PCA and t-SNE [132].

4.3 Evaluation

In this section, we first give a detailed analysis of our method compared to the baseline and discuss several aspects of our method. We start with the influence of the network architecture on the methods performance. Further, we analyze the scalability of our method in term of computational time and provide ablation studies on the hyper-parameter λ, that balances the influence of each task during training as well as the input modality used to train our model and the effect of rotation parameterization used to describe the object's pose.

4.3.1 Comparison to the Baseline Methods

We first evaluated our method on models trained on five, ten and fifteen objects, for which the mean angular error is reported in Table 4.1. Note that for nearest neighbor pose retrieval only the poses of correctly classified objects are considered during the evaluation. In comparison for regression the whole test set is used, as in this case features are not trained to directly infer the object's class.

Tab. 4.1. Angular error of the baseline method (*NN*), direct orientation regression (*R*) and our multi-task learning approach (*Rours, NNours*) as well as classification accuracy. The results indicate that a model trained to perform recognition as well as regression benefits from both tasks, which is reflected in its performance.

	15 Objects		10 Objects		5 Objects	
	Mean (Median) ± Std	**Class. Acc.**	**Mean (Median) ± Std**	**Class. Acc.**	**Mean (Median) ± Std**	**Class Acc.**
***NN* [236]**	25.3° (11.8°) ± 40.8°	92.5%	19.9° (**10.6°**) ± 34.8°	92.6%	24.2° (10.7°) ± 43.3°	99.3%
NNours	**17.7° (11.6°) ± 25.8°**	**97.1%**	**14.7°** (11.5°) ± **15.0°**	**97.5%**	**13.1° (10.3°) ± 15.2°**	**99.9%**
R	38.2° (26.2°) ± 34.7°	-	29.2° (20.7°) ± 28.0°	-	22.1° (15.6°) ± 24.4°	-
Rours	**27.3° (19.3°) ± 27.3°**	-	**23.1° (17.6°) ± 21.3°**	-	**19.2° (13.8°) ± 21.5°**	-

Overall, we were able to obtain a significant improvement in performance for both regression as well as nearest neighbor search accuracy by our proposed method. During training, the usually more difficult regression task, seems to be easier for the network to achieve when provided with a meaningful embedding. As a result the mean angular error is improved by 28.8% in comparison to a model trained purely on regression. Since poses as well as objects are already well-distinguished and the feature descriptors separated by the triplets and pair loss functions, the regression can leverage the information contained in the features and, in turn, more easily be learned.

As for the performance of nearest neighbor search we found an improvement in robustness and accuracy of our multi-task learning framework compared to the baseline as well. The standard deviation as well as the mean angular error of our model, *NNours*, decreases significantly, making the method more robust. Here we can report a relative improvement of 30.0% for the mean angular error while training on the full LineMOD dataset, meaning fifteen objects.

Both methods, regression and nearest neighbor retrieval, benefit from jointly learning robust features and poses. The choice of which model to choose now becomes a trade-off between time complexity and accuracy, which we will address further in section 4.3.5.

Next, to analyze our resulting feature descriptors, we compare our method using nearest neighbor search, *NNours* to the baseline method, for which we report the classification and pose accuracy in Table 4.2. Again, our experimental results show that the feature descriptors provided by the model trained on both tasks seem to be better suited for the overall aim. As a result the nearest neighbor pose retrieval accuracy improves upon the baseline method. Additionally, we are able to improve upon the classification accuracy compared to the state-of-the art method.

4.3.2 Qualitative Evaluation

Figure 4.5 shows a qualitative comparison between the top nearest neighbors retrieved by our method when trained and evaluated on depth information. We show results on ten objects of the LineMOD dataset, where the first row corresponds to the query image and subsequent rows show the top k-th retrieved neighbor. Based on the results, one can see that our method is well able to retrieve correct object classes as well as corresponding viewpoints to the one of the query image patch.

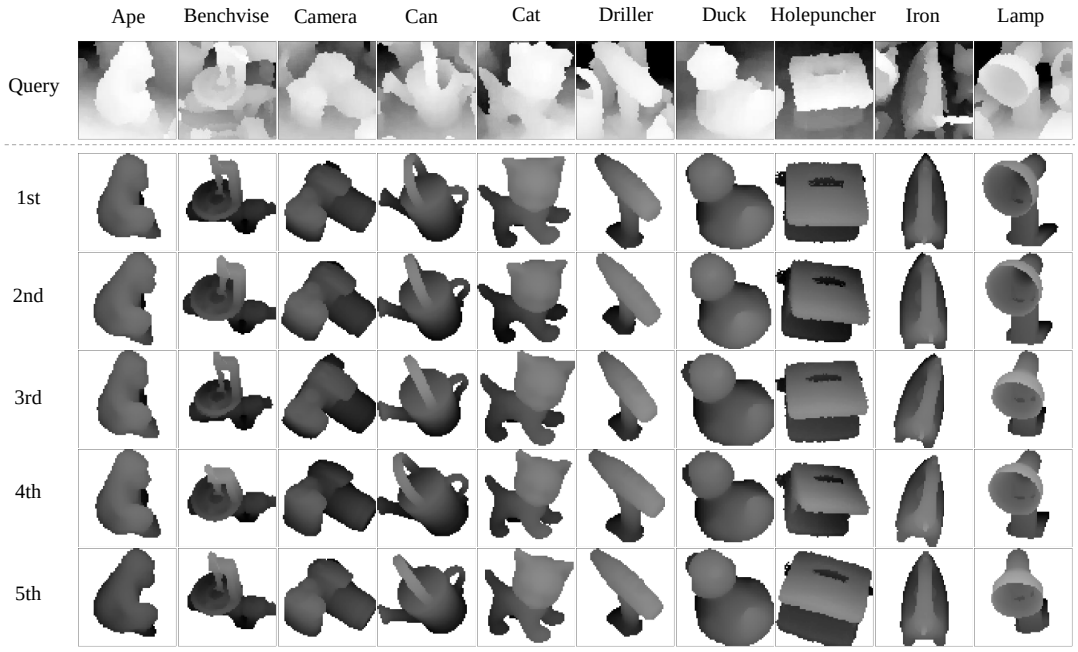


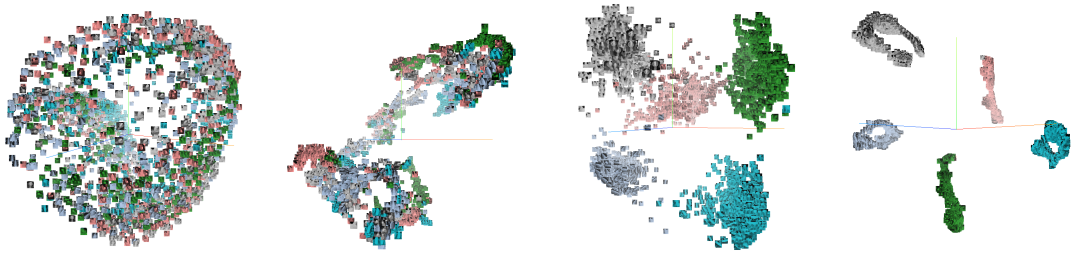
Fig. 4.5. Query patch and its top five retrieved nearest neighbors from the database obtained from our method *NNours* when trained and evaluated on depth information.

	Angular error			Classification
	10°	20°	40°	
NN [236]	35.98%	71.56%	82.72%	92.46%
NNours	37.89%	79.61%	92.27%	97.07%
NNours$_{deeper}$	**41.32%**	**82.52%**	**93.51%**	**97.26%**

Tab. 4.2. Influence of the network architecture on the performance of our method. A comparison between the classification and angular accuracy of the baseline method, *NN*, and our results on fifteen objects of the LineMod dataset is reported.

4.3.3 Influence of Network Architecture

To explore the methods performance with respect to network architectures with varying depths, we additionally run our model using the network architecture described in [32]. This architecture in comparison to the one used so far adds two more convolutional layers to the network and removes max pooling layers by including convolutional layers with stride two. Stated by the authors of [226], a deeper network architecture did not seem to improve the accuracy of the method significantly. While for our baseline model we experienced the same behavior, by using our multi-task learning framework and testing on a deeper network architecture we found that we can improve the pose estimation accuracy even further. Here we were able to achieve the results seen in Table 4.2, abbreviated as *NNours*$_{deeper}$. We can report a relative improvement of 7.2% using nearest neighbor search and 9.0% in the mean angular error of our regression results by using a deeper network architecture. Results are reported while training on the full LineMOD dataset of fifteen objects. However, further experiments are necessary to analyze and optimize the network and assess the effect on the regression accuracy with respect to the network architecture.



(a) Obtained using direct pose regression. **(b)** Obtained using our multi-task learning framework.

Fig. 4.6. Feature visualization using left: PCA and right: t-SNE [132] for five objects of the LineMOD [90] dataset. Object classes are used for color coding. By using a multi-task learning framework, we are able to improve feature descriptors learned for object pose estimation resulting in a well clustered feature representation.

4.3.4 Feature Visualization

The resulting improvement of our method in comparison to the baseline stems from a better suited feature representation learned. Therefore, to qualitatively assess the improvement in accuracy for both pose regression and nearest neighbor pose retrieval, we visualize the learned features for both pure pose regression and in comparison our multi-task learning framework. Nevertheless, the learned features live in a high-dimensional space that is not easy for us to understand. For a clear visualization, we therefore map the high-dimensional features to a 2D or 3D space. For this purpose, we use PCA and t-SNE to visualize the descriptors as both methods are suitable to map the dimensionality of the descriptors into the lower dimensional 3D-space.

We use TensorBoard [134] to create the feature visualizations. For t-SNE, we use a perplexity of 100 and a learning rate of 10 until convergence. When using PCA, the variance including the best three components resulted in 53.2%. The resulting clusters for five objects can be seen in Figure 4.6, showing that features belonging to the same object class are nicely clustered by our method. In comparison the features extracted by a direct pose regression method naturally show no clear distinction between classes as this information is not the aim of such methods.

4.3.5 Scalability

Our proposed method, to some extent, can be trained on multiple objects considering some loss in accuracy in pose estimation. However, with an increasing number of objects, nearest neighbor search shows a natural increase in computational time as well. Therefore, in this section we analyze the time complexity and accuracy of our models when trained for different numbers of objects. For nearest neighbor search we use a standard OpenCV matcher. Figure 4.7 shows the mean time of our models and the corresponding angular error. The mean time for regression is calculated as one forward pass of the neural network. For nearest neighbor methods only the matching time is reported. To obtain the total computational time required, the constant time of one forward pass should be added to the shown results for matching. One

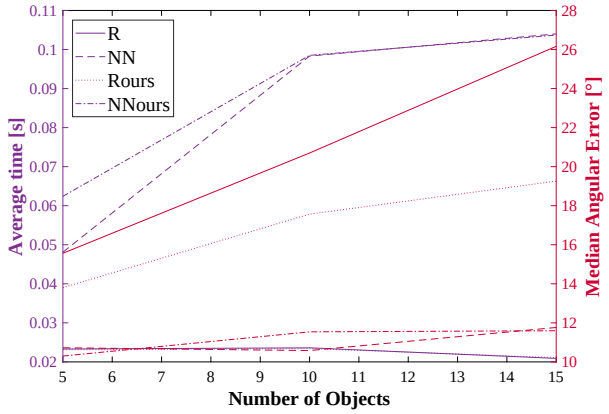


Fig. 4.7. Average time and median angular error of nearest neighbor pose retrieval, regression and our approach evaluated on the LineMOD dataset.

can see nicely that regression has a constant time, regardless of how many objects are used, whereas for nearest neighbor searches the time increases with additional objects. Depending on the application, this and the drop in accuracy with increasing number of objects should be taken into account.

4.3.6 Sensitivity to Regularization Parameter λ

Since our loss function includes a regularization parameter λ balancing the two components of regression and manifold learning, we conducted experiments on the sensitivity of this parameter using ten objects of the LineMOD dataset. By choosing different values for λ and thus weighing either the $\mathcal{L}_d$ loss or the pose loss $\mathcal{L}_{pose}$ more, we found that the results improve for nearest neighbor pose retrieval, if the two terms are equally weighted, and decreases when focusing more on the regression loss. Regarding regression, we observed similar results: improvement when additionally focusing on the $\mathcal{L}_d$ loss, enhancing the feature representation and decrease, if the model is only trained on regression. Nevertheless, it can be seen that regression has a much stronger influence on the nearest neighbor pose retrieval in terms of performance than the other way around. Thus, to obtain an optimal balance, we set $\lambda = 0.5$ in our experiments.

The results, depicted in Figure 4.8, emphasize our assumption that the two terms are beneficial to one another, i.e. both features and pose regression are mutually optimized. Note that we omit the regression result in case the model was only trained on the feature representation, since the regression layer in this case is not included in training and will not provide meaningful results.

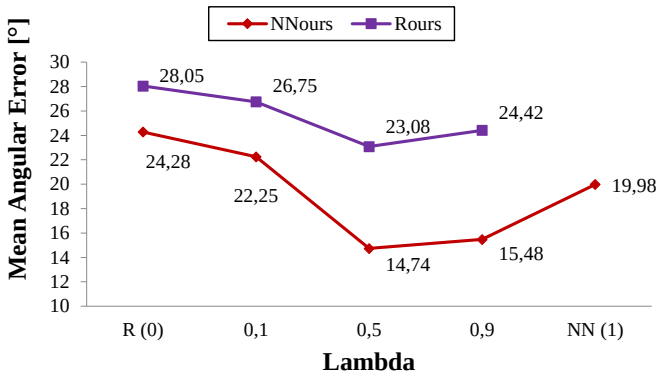


Fig. 4.8. Sensitivity of λ in our loss function $\mathcal{L}_{MTL} = (1-\lambda)\mathcal{L}_{pose} + \lambda\mathcal{L}_d$. Depicted is the influence of different weighting parameters on the mean angular error for regression as well as nearest neighbor pose retrieval.

4.3.7 Effect of the Input Modality

We evaluate in influence of the chosen input modality on the performance of the baseline and our method. We compare between RGB, Depth and RGB-D image patches that are used as an input to the neural network. In terms of classification accuracy the combination of RGB and Depth information seems to be performing the best. When only relying on RGB information, objects can be easily misclassified due to similar color and appearance, which is also visible in Figure 4.9, where we show the five top nearest neighbors for an example query image obtained by our method when relying only on RGB information. Due to similar colors and shape resulting from certain viewpoints, the iron object is misclassified and predicted as the benchvise object. Therefore, depth information can aid in such cases and provide additional shape information. The accuracy of predicted poses, however, strongly depends on the input modality, although only relying on RGB information seems to perform worse for all methods.

Tab. 4.3. Angular error of the baseline method (*NN*), regression (*R*) and our approach (*Rours, NNours*) for different input modalities, i.e. only RGB, only depth or both, RGB-D. We report the results on our subset of 10 objects from the LineMOD dataset.

	RGB		Depth		RGB-D	
	Mean (Median) ± Std	**Class. Acc.**	**Mean (Median) ± Std**	**Class. Acc.**	**Mean (Median) ± Std**	**Class. Acc.**
***NN* [236]**	24.2° **(14.6°)** ± 31.8°	97.0%	20.0° **(10.6°)** ± 34.8°	92.6%	22.7° **(13.8°)** ± 31.8°	98.4%
NNours	**20.9°** (14.7°) ± **22.5°**	**97.8%**	**14.7°** (11.5°) ± **15.0°**	**97.5%**	**18.3°** (13.6°) ± **18.8°**	**98.9%**
R	**25.7°** **(15.8°)** ± 28.1°	-	29.2° (20.7°) ± 28.0°	-	25.3° **(16.1°)** ± 28.3°	-
Rours	27.5° (18.5°) ± **28.0°**	-	**23.1°** **(17.6°)** ± **21.3°**	-	**23.5°** (16.6°) ± **24.0°**	-

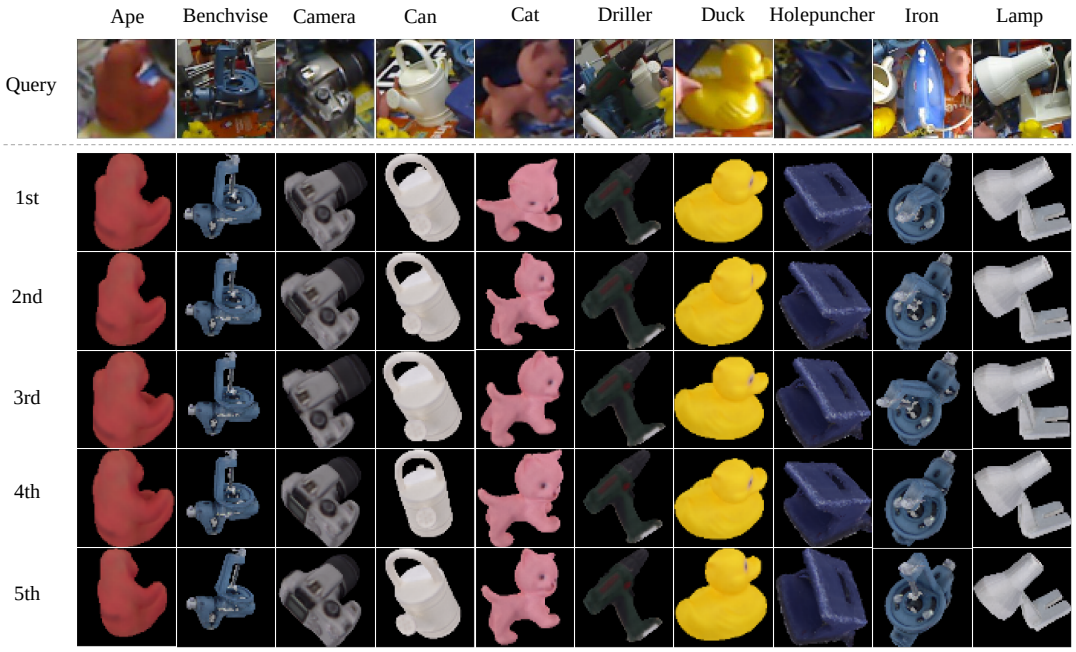


Fig. 4.9. Query patch and its top five retrieved nearest neighbors from the database obtained by our method *NNours* when trained on RGB information.

4.3.8 Influence of Rotation Parameterization

Further, we study the effect of different rotation parameterizations on the performance of our model. In particular we compare to the recently proposed continuous six dimensional representation of Zhou et al. [247]. In their work the authors address the issues of various rotation parameterizations when used to train neural networks and elaborate on which representation is best suited for such a task. Further the authors propose a continuous representation and show the advantages such a representation has when training neural networks in comparison for example to euler angles or rotation matrices that are not continuous. In our application, we observe similar performance in nearest neighbor retrieval and an improved performance in regression accuracy, which we summarize in Table 4.4. This leads us to assume that the different parameterization does not highly affect the properties of the learned features, however, the improved regression performance stems from the in general better suited representation for rotation learning with neural networks.

Tab. 4.4. Angular error of our approach (*Rours, NNours*) for different rotation parameterizations. We report the results on our subset of ten objects from the LineMOD dataset.

Parameterization	***Method***	**Mean**	**Median**	**± Std**	**Classification**
Quaternion	***NNours***	14.7°	11.5°	± 15.0°	97.5%
	Rours	23.1°	17.6°	± 21.3°	-
6D	***NNours***	15.8°	12.4°	± 15.4°	95.3%
	Rours	20.2°	16.3°	± 16.8°	-

4.3.9 Remarks on Object Pose Estimation

Before moving to the next application, we now briefly summarize the findings and conclusions of the presented method. In the previous sections we have described a multi-task learning framework for object recognition and pose estimation. Once determined the results can aid in a number of applications, such as navigation in robotic grasping. By introducing a novel loss function, combining regression and manifold learning, we were able to improve both direct pose regression as well as nearest neighbor pose retrieval by a large margin, compared to the baseline methods. Thus, we conducted a detailed analysis of feature descriptor learning, direct regression and the effect that both tasks have on each other in the context of object pose estimation.

Using our proposed loss we were able to improve the pose regression performance and robustness with respect to only using a direct regression model. Additionally, the viewpoint descriptors learned with this loss seem to be much more discriminative compared to those learned with the triplet loss by itself. Therefore, when we use them for the nearest neighbor search we get the best performance. Pose regression seems to be more difficult problem to solve alone, but when applied on a manifold learned features, where discrete poses are already well separated it becomes easier to perform pose regression and achieve better performance than by standalone direct pose regression.

While we were able to improve the regression performance using our multi-task learning framework, its accuracy remain inferior to the nearest neighbor pose retrieval. This overall difference between the regression accuracy and the improved pose retrieval, based on nearest neighbor search, might partially be due to the non-constrained space on which the poses are predicted. In our training set, we constrain the poses as well as in-plane rotations to a certain amount of degrees. However, while predicting the pose, no such constraint is enforced by the neural network. By constraining the network during training and enforcing geometric bounds, we should be able to improve the pose regression performance further.

By including real data we were able to match synthetic and real images onto the same domain. Still it might not be possible to fully cover the pose space, which we found to significantly impact the pose regressions performance as some poses might not be mapped sufficiently. It is, however, still possible for the synthetic samples. In the ideal case, only rendered samples should be used in the training set, which would then also result in removing this limitation as additional renderings can easily be created. Future work therefore includes improving our methods generalization capabilities by only using synthetic images for training, and removing the need to rely on real data at all. This can include the use of more sophisticated and extensive data augmentation, which especially considering depth information can become very similar in appearance to real renderings. In terms of RGB information, where illumination, texture and shading significantly changes the appearance of an object in the rendering, this task becomes more challenging. Various methods, however, addressing recent domain adaptation, image synthesis techniques or incorporating physics-based rendering could be applied. We leave this as future work, but believe that such optimization can push the performance of our regression approach beyond the one of nearest neighbor pose retrieval and create a generalized model. As a result we would obtain a model that scales well with the number of objects and in turn optimizes the method's memory consumption and efficiency as well.

4.4 Evaluation on Place Recognition

In our second application, we evaluate our method on the task of large-scale place recognition. Our aim in this case is to retrieve the correct geo-location for a given query RGB image. Our labels, in this case, consist of GPS coordinates of the locations at which an image was taken. We first describe our experimental setup and changes made to adapt our method to the task at hand. We then introduce the used datasets before presenting the results and evaluation of our method for this task.

4.4.1 Experimental Setup

As neither dense depth images nor accurate 3D models of the scene are commonly available in place recognition datasets, we adjust our method to meet the requirements for this particular application. For this aim we utilize RGB instead of depth images as an input and estimate the GPS location of a query image instead of its rotation. Our model, therefore, now aims to estimate the mapping $\mu_{\mathbf{\Gamma}}(\mathbf{X}) : \mathbb{R}^{W \times H \times 3} \mapsto \mathbb{R}^2$, where W and H are the width and height of the RGB image $\mathbf{X}$. To train the parameters of the neural network $\mathbf{\Gamma}$ we remain with our original loss function $\mathcal{L}_{MTL}(\mathbf{\Gamma}|\mathbf{X})$ of Equation 4.2. Given that synthetic renderings are not available, our aim becomes reducing the domain shift between seasonal and illumination variations. For this purpose, we construct triplets as well as pairs accordingly and remain with our triplet and pair-wise loss functions.

Further, we adjust our network architecture. In particular, we employ a ResNet-18 as our backbone network. Due to the absence of a proper model and the possibility to create realistic synthetic renderings on-the-fly during training, we utilize a on ImageNet pre-trained network. We remove the classification layers and add one fully-connected layer with output dimensionality d, which we use to train a robust feature representation. For our regression model we add one additional fully-connected layer with respective dimensionality according to the task, here two, for GPS longitude and latitude values. We augment the training data by applying random brightness changes for further robustness to illumination changes. For this particular application, we set a constant margin of $m = 0.1$.

Datasets

We evaluate our method on the Tokyo 24/7 [216] and the symphony seasons [81] dataset. Both datasets are commonly used for long-term visualization to evaluate place recognition as well as camera localization methods in dynamic and changing environments.

Tokyo 24/7 This dataset consists of images taken in Tokyo at varying times of the day. For each location there are three categories, *daytime*, *sunset* and *night*, during which images are taken. In addition the camera's orientation is changed in each location, such images with very different viewpoints are taken at the same GPS location. Example images of the dataset can be seen in Figure 8.3.

Symphony Seasons The symphony seasons dataset, depicts the shore of Symphony Lake in France. The reference and query images were captured by a camera on an unmanned boat.

Fig. 4.10. Example images of the Tokyo 24/7 dataset. (Columns) At each viewpoint images are taken during daytime, sunset and night. (Rows) Further, at each GPS location the viewpoint of the camera is changed, resulting in highly different images with similar location labels.

The boat is deployed at changing periods of time, capturing one traversal each time with varying seasonal changes and illumination conditions. All images were recorded in sequences. The Symphony Seasons dataset represents an autonomous driving/monitoring scenario, where it is necessary to localize images taken under varying seasonal conditions such as dusk or winter. It further shows little texture and human-made features, which challenges existing localization methods.

4.4.2 Experiments

We first present our results on the Tokyo 24/7 dataset before describing our findings on the symphony seasons dataset, for which we present quantitative as well as qualitative evaluations in comparison to the baseline method.

Evaluation on the Tokyo 24/7 dataset

The Tokyo 24/7 dataset is in the original publication used for validation only. Due to the design of the dataset, however, it perfectly fits the requirements of our method. Therefore, we extract two third of overall 1125 images and use them as our training set. We test on the remaining images, where we choose the *daytime* and *night* images as our training set and test on the *sunset* category. We construct our triplets the following way, we choose *daytime* images as our anchor images and images with the same GPS coordinates as our puller. Pusher images will be chosen as images with GPS coordinates different from the anchor. Note that due to the design of this dataset, the anchor and puller images can have highly different visual information. An example triplet can be found in Figure 4.11. We compare to our baseline method, *NN*, and perform nearest neighbor retrieval using the features learned from the neural

(a) Anchor (b) Puller (c) Pusher

Fig. 4.11. Example Triplet of the Tokyo 24/7 dataset. a) The anchor image depicted at *sunset*, b) shows an image taken at the same location but under very different illumination conditions, i.e. night and daytime, and, c) an image taken at a different GPS location.

Tab. 4.5. Percentage of correctly localized query images for ours and the baseline method on the Tokyo 24/7 dataset. With our learned feature representation we are able to improve upon the baseline method by a large margin.

		kNN					
Method	**d**	**1**	**2**	**5**	**10**	**20**	**50**
NN	64	1.2%	3.5%	7.2%	12.0%	20.7%	43.4%
	512	2.7%	4.5%	9.3%	19.2%	32.0%	59.2%
NNours	64	10.7%	18.4%	36.0%	49.3%	63.7%	82.1%
	512	29.9%	41.6%	57.6%	72.3%	83.5%	94.7%

network trained purely on the triplet loss. We evaluate two versions of our method, where we set the dimensionality of the resulting feature descriptor to $d = 64$ and $d = 512$. We report our results in Table 4.5, showing the percentage of correctly localized frames considering the top k nearest neighbors where $k \in [1, 2, 5, 10, 20, 50]$. As there are only few images in the dataset and multiple viewpoints are covered at the same location we employ a very strict threshold and only consider an image to be correctly matched if the location corresponds exactly to the ground truth one. Again by employing our multi-task learning framework, in this case aiming to regress longitude and latitude of the geo-location, we are able to learn more representative feature representations for the task at hand.

Evaluation on the Symphony Seasons Dataset

For the Symphony Seasons dataset we choose five sequences for training and two for testing, resulting in around 10k and 3.5k images with GPS labels. Each sequence is assigned to a specific weather condition, such as dusk or winter. In this case we consider an image to be matched correctly if the retrieved nearest neighbor position is within a ten meter distance to the ground truth one. Based on this, we form triplets and choose the puller to be within this distance. We randomly assign a pusher image such that its distance is larger than ten meter from the ground truth. Table 4.6 shows the percentage of correctly localized frames of our method and the nearest neighbor baseline, *NN*. The results support our previously made conclusion and show a significant improvement in performance when training with our multi-task learning framework.

Tab. 4.6. Percentage of correctly localized query images for ours and the baseline method on the Symphony Seasons dataset. A localized image is considered correct if the predicted location is within 10 meters of the ground truth.

		kNN					
Method	**d**	**1**	**2**	**5**	**10**	**20**	**50**
NN	512	55.4%	64.3%	71.4%	75.0%	78.3%	81.8%
NNours	512	69.9%	76.0%	81.3%	83.3%	85.0%	87.2%

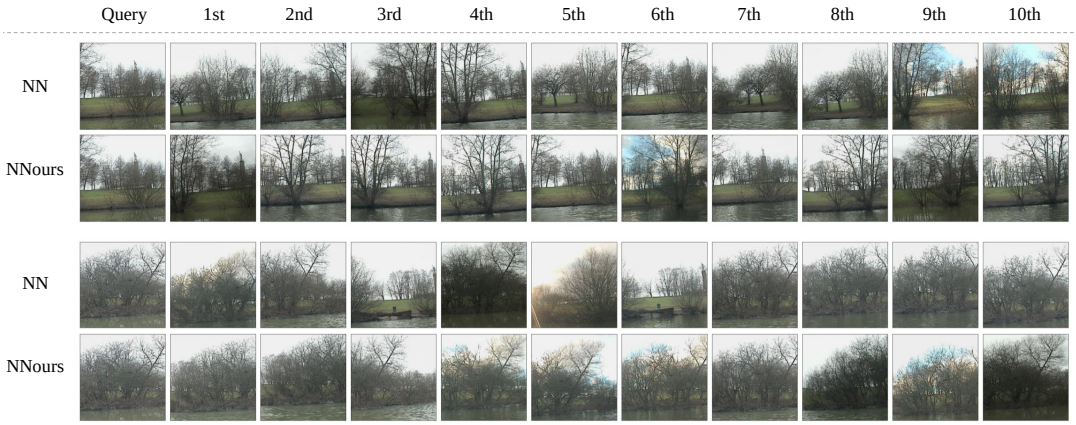


Fig. 4.12. Query patch and its top ten retrieved nearest neighbors from the database obtained by our method *NNours* and the baseline *NN*.

Further, we show qualitative result of the baseline, *NN*, and our method in Figure 4.12. In this case, we depict the query image in the first column and show the corresponding retrieved k-th nearest neighbor in the remaining columns. As one can see, the dataset contains challenging views as the variation in vegetation or occurrence of distinguishable landmarks is limited. Further, the changing weather and illumination conditions pose an additional challenge. Nevertheless, in comparison to the baseline, our method is able to capture small details that correctly reflect the position of the query image and retrieve an appropriate nearest neighbor view.

Remarks and Future Work

Although our method is showing promising results on the task of place recognition, the construction of triplets and pairs that our method relies on poses an additional challenge. As synthetic renderings are difficult to obtain for this particular application, domain changes mostly occur due to seasonal and illumination variation. However, obtaining such datasets with exact viewpoints might not be feasible. Analyzing and incorporating approaches that aim to solve the domain shift problem without relying on available ground truth pairs into our method would therefore be of high interest and value. We leave this investigation to future work.

4.5 Conclusion

We have presented a general framework for robust feature learning by leveraging a multi-task learning framework. By incorporating manifold learning as well as pose/location regression, we have shown that in comparison to a baseline image retrieval method, our framework is well able to leverage the viewpoint information for better retrieval performance and have demonstrated the effectiveness of our method on object recognition and pose estimation as well as large-scale and long-term place recognition.

However, nearest neighbor retrieval methods are naturally restricted to the information contained in the given database. Therefore, in the next chapters, we move on to methods that entirely rely on regression and, theoretically, can output the entire range of possible values for a given task at hand. Moving from place recognition, we evaluate these methods on camera localization, predicting the six degrees of a camera pose instead of relying on coarse GPS information.

Part III

Structure-Based Pose Estimation

5 Introduction

5.1 Motivation

In classical simultaneous localization and mapping or structure from motion (SfM) scenarios the camera is tracked in an unknown environment and a corresponding sparse 3D map or reconstruction of the environment is built. Each 3D point in the map has a corresponding image feature descriptor associated to it. Therefore, the main component of these methods is the detection of key-points in a query image and feature extraction, e.g. SIFT features, at these respective points. 2D to 2D point correspondences between pairs of images can be established using feature matching and used to compute the relative camera motion between frames. Inferring the relationship between consecutive frame pairs constructs a graph of viewpoints and a 3D map of the environment can be reconstructed via triangulation. Finally, bundle adjustment can be used to jointly refine the reconstruction as well as inferred camera parameters.

Once established, the constructed map defines the coordinate frame of the scene and 2D to 3D point correspondences between the image and the 3D model can be established in a similar fashion as earlier described using feature descriptor matching. Finally, given these correspondences, depending on the application the absolute camera pose can be computed by solving the perspective-n-point problem and used, e.g. for re-localization in the global map.

Due to the rich geometric information available these methods usually provide accurate camera localization accuracy, however, can easily fail in case of texture-less surfaces as a result of the absence of significant features. Moreover, efficient feature matching techniques are required to achieve reasonable computational times for camera re-localization applications. To address this issue, in the following, we explore methods that do not require explicit feature matching but are designed to implicitly find corresponding points such that the initially required matching step can be bypassed efficiently.

Additionally, outliers, which we illustrate in Figure 5.1, in the established correspondences most often hinder a direct computation of an accurate camera pose estimate. A most prominent solution proposed is RANSAC, repeatedly sampling a minimal subset of correspondences to compute a camera pose and finally choosing the hypothesis with the largest inlier set. Although able to find a robust solution, the probability of finding such a solution significantly decreases with the number of outliers and correspondence quality, resulting in high computational times to ensure an accurate estimate is found. Instead, we aim to directly infer the quality of found point correspondences by regressing confidences for each correspondence depending on their accuracy. This way, based on their confidence values, erroneous predictions, or outliers, can be easily identified.

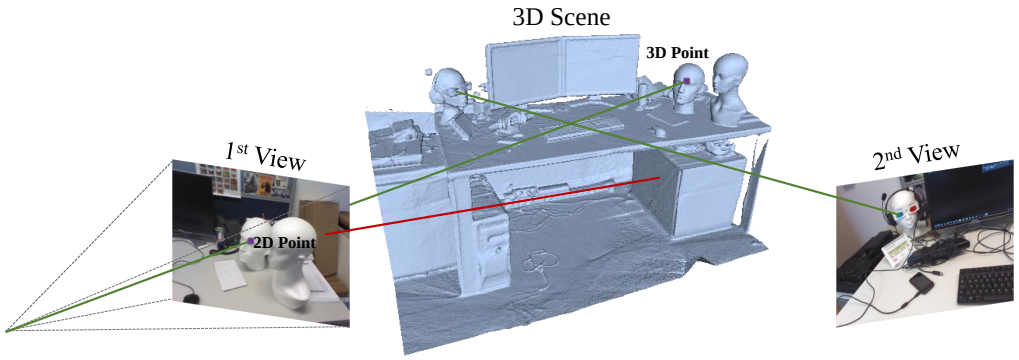


Fig. 5.1. Structure-based methods rely on point correspondences between a 2D image view and the 3D scene to estimate the corresponding camera pose. Correct matches, most commonly called inliers (shown in green) can provide highly accurate camera pose estimates, whereas outlier (depicted in red) hinder such computation.

5.2 Related Work

In this context, related work on structure-based methods can be divided into three main directions. A first category includes *feature matching* methods, which rely on the traditional pipeline of extracting features, matching them to the 3D model and subsequent pose estimation [155, 190, 191, 192, 197, 215]. In the context of recent developments in deep learning each step of the pipeline has been analyzed and, if sensible, their traditional counterparts exchanged or improved with learned methods of machine or deep learning. As such, we consider *scene coordinate regression forests* as another main direction, in which matches are implicitly given by the trained forest [41, 42, 86, 199, 220]. Further, *deep learning based methods* quickly emerged, in addition relying on predicting 3D coordinates, so called scene coordinates, with no necessity of an additional matching step [21, 24, 40].

Feature Matching For this purpose, Sattler et al. [190, 191, 192] propose an optimized prioritization scheme based on vocabulary-based quantization for efficient feature matching. Additionally, by using co-visibility constraints or semantic consistency checks [215], wrong matches can be removed, which further improves the methods accuracy.

In contrast, Schmidt et al. [197] focus on optimizing extracted features used for correspondence matching. Here, a deep learning method is applied and a neural network is trained on a contrastive loss function, pushing features of pixels to be similar only if they correspond to the same 3D point.

D2-Net [61] jointly learns feature descriptors as well as key-points, replacing traditional hand-crafted feature extraction and key-point detection. A 3D feature map learned by a convolutional neural network is interpreted as an attention map, where the maximum of a particular descriptor over the feature dimensionality is considered a key-point at a specific pixel location if that pixel location at the same time corresponds to a local maximum in the 2D feature map. At the same time, feature descriptors can be used to find corresponding matches between images and are trained by a triplet loss to handle strong illumination and viewpoint

changes. An additional loss term is added for detecting key-points such that distinctive correspondences obtain high attention scores.

In [71] the authors Geppert et al. propose an efficient matching for 2D-3D correspondences in a multi-camera setting. Given the additional information obtained from multiple cameras, efficient strategies to find plausible matches become increasingly important. The method builds upon Active Search, however, adds an extra cost term for prioritization matching that enforces the feature distribution to be spread amongst the cameras. Pose priors are additionally included to obtain robust and reliable pose estimates and additional constraints of visual odometry measurements included to enable drift-free localization.

Scene Coordinate Regression Forests Implicitly giving a mapping between image pixels and 3D points without the need for feature matching, Shotton et al. [199] train a regression forest on RGB and depth features extracted at pixel locations to estimate the corresponding scene coordinate directly. All coordinates in a leaf are then clustered using mean shift and the mode of the largest cluster used as a label scene coordinate for that leaf. A pose is obtained by back-projecting the image pixels their depth information and the camera's parameter and then applying Kabsch algorithm on the 3D camera coordinates and predicted 3D scene coordinates. However, as predictions can be noisy and contain a lot of outliers, a pre-emptive RANSAC is applied to obtain a robust and accurate solution. Further extensions and analysis of this method have been proposed, including backtracking schemes [145], comparison to neural networks [140] and uncertainty of the forests predictions [220]. In this work, in the leaves of the regression forest anisotropic Gaussian mixture models are fitted over the scene coordinate clusters instead of obtaining a single mode. This allows for uncertainty estimation over the predictions as well as continuous pose optimization.

Additional information in the form of line segments is used in [146] by Meng et al. to further remove outliers. Line segments are extracted from the RGB image and used to sample points. Further outliers are removed by backprojecting the sampled points and fitting a 3D line with RANSAC.

An ensemble prediction of regression forests is proposed in [86] by Guzman-Rivera et al. predicting multiple pose hypothesis, clustering them and then choosing the best one according to a scoring function dependent on the 3D model. In addition rendered views of the pose hypothesis are used to further refine the final predicted camera pose.

As of now, most presented methods are trained scene-specific and can therefore only be applied in that, assumed to be static, scene. Therefore an online adaption method of the regression forest has been proposed in [42] by Cavallari et al. While training is still scene-specific, the model can be adapted to any new scene within a couple of query frames. For this aim, assuming the camera poses of these frames are correctly tracked, the frames are passed to the regression forest, the distribution at the leaves of the forest from the old scene discarded and updated with the 3D coordinates of the new scene. The authors extend their work in [41] for further optimization of the method within a relocalization cascade while remaining real-time performance. All the above methods, however, heavily rely on 3D or depth information.

Scene Coordinate Neural Networks Switching from regression forests to convolutional neural networks, Brachmann et al. [21] propose an end-to-end trainable pipeline, consisting of a scene coordinate regression and a pose hypothesis scoring network, connected by a differentiable version of RANSAC, which they call DSAC. A hypothesis is then chosen by a probabilistic selection, making RANSAC differentiable and enabling end-to-end training with neural networks. The method shows remarkable results in retrieving accurate camera poses, however, still requires depth information or a 3D model. Therefore, the authors propose DSAC++ [24], assuming a constant scene-dependent depth prior to initialize the training procedure after which accurate coordinates can be learned optimizing for the re-projection error.

Although, showing good performance, generalization to large-scale scenes of the method still remains challenging. Therefore, a line of research followed, utilizing scene coordinate regression with neural networks [23, 25, 30, 244]. For instance, Budvytis et al. [30] propose to jointly learn local object scene coordinates and global instance labels. Based on the semantic information normalized scene coordinates are learned enabling city-scale scene coordinate regression.

Similar to [199, 220] the aforementioned methods are scene-specific, therefore, Cavallari et al. [40] propose a method to leverage the already trained network to obtain predictions for a new scene in an online fashion. First, a regular grid is placed on the training scene, such that for each predicted 3D coordinate a grid cell index can be computed. Each grid cell in turn storing the predicted scene coordinates falling into that grid. For any new scene, given a couple of frames, the database can be filled with scene coordinates of the new scene online. Similar to the leaves of a regression forest, coordinates contained in each cell can be clustered and their covariance used for robust camera pose estimation with pre-emptive RANSAC, pose refinement on the inlier correspondences and final refinement using ICP.

The work presented in this thesis falls into the category of scene coordinate regression with neural networks. However, compared to the recent methods, which usually employ RANSAC to obtain a robust pose estimate from the established point correspondences, we propose to regress confidences of these correspondences, which allows us to immediately discard erroneous predictions and improve the initial pose estimates. Finally, the resulting confidences can be used to score initial pose hypothesis and aid in pose refinement, offering a generalized solution to solve this task.

6 Coordinate Regression with Confidence Learning

Scene coordinate regression is becoming a highly researched topic in the computer vision community and an essential part of current camera relocalization methods. Different versions, such as regression forests and deep learning methods, have been successfully applied to estimate the corresponding camera pose given a single input image. In this work, we propose to regress the scene coordinates pixel-wise for a given RGB image by using deep learning. For this aim, we first introduce our methodology developed within the scope of this thesis and presented in large portions in the publication

'Scene Coordinate and Correspondence Learning for Image-Based Localization' by Mai Bui, Shadi Albarqouni, Slobodan Ilic and Nassir Navab, published in Proceedings of the British Machine Vision Conference (BMVC), 2018 [31].

We then present evaluations of said method, ablation studies and a comparison to the state of the art on publicly available benchmark datasets for camera re-localization.

6.1 Methodology

In comparison to the method presented in the previous chapters, that has focused on predicting an object's orientation or geo-position of an image, our aim is now to estimate the full six degrees of freedom for a camera pose in reference to a given scene. Therefore, given an input RGB image $\mathbf{X} \in \mathbb{R}^{W \times H \times 3}$ of a scene, where H and W are the image height and width, respectively, our goal is to estimate the corresponding camera pose, given by its orientation $\mathbf{R} \in \mathbb{R}^{3\times3}$ and position $\mathbf{t} \in \mathbb{R}^3$. As described in Chapter 2 the camera pose describes the mapping between the camera, $\mathbf{x}_c$, and the scene coordinates, $\mathbf{X}_w \in \mathbb{R}^3$, as

$$\mathbf{x}_c = \mathbf{R}\mathbf{X}_w + t. \tag{6.1}$$

The relation between the 3D camera coordinates $\mathbf{X}_w \in \mathbb{R}^3$ and the image pixels $\mathbf{p}_x \in \mathbb{R}^2$ depends on the camera's focal length f_x, f_y and the optical center c_x, c_y, and is defined as

$$\mathbf{p}_x = (\frac{f_x x}{z} + c_x, \frac{f_y y}{z} + c_y)^T, \tag{6.2}$$

with $\mathbf{x}_c = (x, y, z)^T$ being a point in the camera coordinate frame, given by its coordinates x, y and its depth value z. In case the camera pose is unknown, it can be retrieved given the

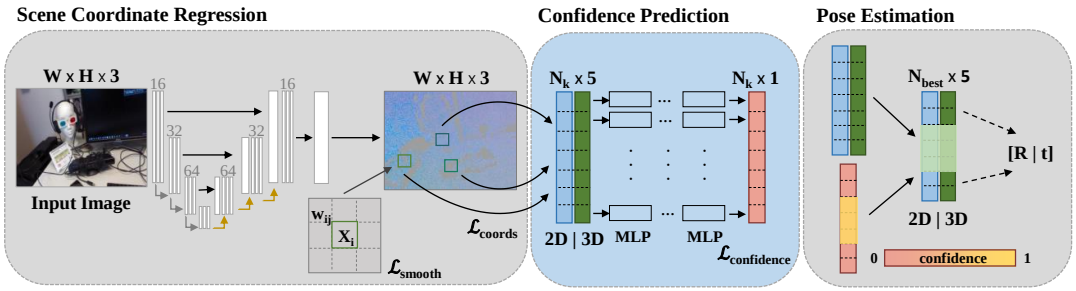


Fig. 6.1. Outline of our re-localization framework. Scene coordinates, $\mathbf{X}_w$, are densely regressed for each pixel in the input RGB image. Confidences, δ, are predicted for a subset of the point correspondences and finally used to compute the camera pose estimate.

minimal number of required correspondences. In case of 3D-3D correspondences between $\mathbf{x}_c$ and $\mathbf{X}_w$, assuming that depth information is available, the camera pose can be retrieved using the Kabsch, or sometimes called Procrustes algorithm. Solving the perspective-n-point problem, e.g. using a PnP solver such as EPnP [122], in turn provides the camera pose from the 2D-3D correspondences between the image points $\mathbf{p}_x$ and $\mathbf{X}_w$.

The method proposed in this chapter focuses on retrieving the camera pose from these two scenarios. In comparison to feature-based methods, however, we provide an implicit correspondence matching using neural networks. For this aim, we propose a learning-based solution that consists of three steps: (1) *scene coordinate regression*, in which we densely predict scene coordinates, and in this context, add a novel regularization, (2) *confidence prediction*, in which we aim to estimate the quality of our coordinate predictions, and (3) *pose estimation*, in which we employ the aforementioned algorithms to compute the camera pose estimate for the most confident predictions N_{best} categorized by our model in step (2). An overview of our framework is given in Figure 6.1, in which steps (1) and (2) are implemented by neural networks.

6.1.1 Scene Coordinate Regression

In the following each step of our framework is described in more detail, starting with scene coordinate regression with convolutional neural networks. Our regression model is trained with two loss functions, the main *coordinate regression* loss and a *coordinate smoothing* loss, acting as regularization on our model.

Coordinate regression. As the first step, our aim is to model the function $\mu_{\mathbf{\Gamma}}(\mathbf{X}) : \mathbb{R}^{W \times H \times 3} \mapsto \mathbb{R}^{W \times H \times 3}$, obtaining the predicted scene coordinates $\hat{\mathbf{X}}_w$, as $\hat{\mathbf{X}}_w = \mu_{\mathbf{\Gamma}}(\mathbf{X})$, where $\mathbf{\Gamma}$ are the model parameters. Therefore, as ground truth scene coordinates $\mathbf{X}_w$ we can either rely on a reconstruction, for instance computed using structure from motion, or infer the coordinates from the corresponding depth information by backprojecting the 2D pixels according to the camera parameters and ground truth pose. The ground truth is then used to train a model and regress the scene coordinates of each pixel in the RGB input image, in which case we obtain

an output map of $\mathbb{R}^{W\times H\times 3}$ from our model. For this purpose, we use the Tukey's Biweight loss function [9] to regress the 3D coordinates, given as

$$\mathcal{L}_{coords}(\mathbf{\Gamma}|\mathbf{X}) = \frac{1}{W\cdot H}\sum_{i=1}^{W\cdot H}\sum_{s=1}^{S}\rho(r_{i,s})\ , \tag{6.3}$$

$$\text{with } \rho(r_{i,s}) = \begin{cases} \frac{c^2}{6}[1-(1-(\frac{r_{i,s}}{c})^2)^3], & \text{if } |r_{i,s}| \leq c \\ \frac{c^2}{6}, & \text{otherwise} \end{cases} \tag{6.4}$$

where $r_{i,s} = X_w^{i,s} - \hat{X}_w^{i,s}$ is the residual, $S = 3$ is the number of coordinates to regress and $\rho(\cdot)$ is Tukey's Biweight function. In Tukey's Biweight function the choice of the tuning constant c plays a crucial role, which is proposed to be chosen according to the median absolute deviation over the residuals assuming a Gaussian distribution. Nevertheless, we propose to choose the parameter c depending on the spatial extent of the current scene, where, after empirical evaluation, we found half of the scenes diameter given in meters to provide consistent results. In case of missing depth values and thus missing ground truth scene coordinates, we omit these pixels during training in order not to negatively influence the network.

Coordinate smoothing. The graph Laplacian regularization, has been successfully applied for image denoising on image patches [162] and is computed on a neighborhood of pixels, motivating us to consider the neighborhood of a given scene coordinates. Minimizing the graph Laplacian regularizer enables us to smooth the image patches with respect to the given graph. Similarly, we consider the scene coordinates as vertices and compute weights according to the depth value at the corresponding pixel. Thus, we apply a local smoothing term on this neighborhood during training, to ensure similarity between neighboring points. Since this assumption does not hold true in case of edges where large depth differences occur, we weigh each point in a surrounding neighborhood based on their depth difference given by

$$w_{ij} = \frac{e^{-|d_i - d_j|}}{\sum_{k\in K, k\neq i} e^{-|d_i - d_k|}}, \tag{6.5}$$

where, given a pixel position i, we compute weights w_{ij} for each index j in a given neighborhood K. In this case, d_i represent the depth value at index i. Finally, we obtain the additional smoothing term in our loss function

$$\mathcal{L}_{smooth}(\mathbf{\Gamma}|\mathbf{X}) = \sum_{i}^{N_k}\sum_{j\in K} w_{ij}\cdot\|\hat{\mathbf{X}}_w^i - \hat{\mathbf{X}}_w^j\|_2, \tag{6.6}$$

where $\hat{\mathbf{X}}_w$ corresponds to the predicted scene coordinates at a given pixel index. This term loosely pushes the surrounding points closer together, given the fact that their depth values are similar; otherwise, a larger difference between points is accepted.

Overall, we train the model using our loss function, described as $\mathcal{L}(\mathbf{\Gamma}|\mathbf{X}) = \mathcal{L}_{coords}(\mathbf{\Gamma}|\mathbf{X}) + \mathcal{L}_{smooth}(\mathbf{\Gamma}|\mathbf{X})$.

6.1.2 Confidence Prediction

The regressed coordinates can be used to obtain a pose estimate. However, these correspondences usually include a large amount of erroneous correspondences, which can be addressed using RANSAC.
In order not to restrict ourselves in terms of error given by an inlier definition and inspired by [234], which classifies point correspondences between image pairs, we train a neural network to estimate the probabilities of 2D-3D correspondence to directly obtain a measure of the quality of our predicted correspondences. Instead of solving this as a classification problem as in [234], we consider this task as a regression problem. Therefore, we use the output of the model described in the previous section to create probabilities for each scene coordinate prediction and construct the training set $S_{confidence} = \{(\mathbf{p}_{\hat{\mathbf{X}}_w^1}, \hat{\mathbf{X}}_w^1, \delta_1), .., (\mathbf{p}_{\hat{\mathbf{X}}_w^{N_k}}, \hat{\mathbf{X}}_w^{N_k}, \delta_{N_k})\}$, where

$$\delta_i = e^{-(s \cdot \|\mathbf{X}_w^i - \hat{\mathbf{X}_w^i}\|_2)}. \tag{6.7}$$

Here, s is used as a scale, such that accurate coordinates are given a high probability. In this step, the objective is to compute the function $\mu_{\mathbf{\Omega}} : \mathbb{R}^5 \mapsto \mathbb{R}^1$, described by the model parameters $\mathbf{\Omega}$, so that $\delta = \mu_{\mathbf{\Omega}}(\mathbf{p}_{\hat{\mathbf{X}}_w}, \hat{\mathbf{X}}_w)$. To this end, we feed N_k points containing the image pixel and the predicted scene coordinates to our model. As an output, we obtain a probability for each point according to whether it is likely to be a good correspondence or not. As a loss function, we use the l_2 loss to train this model,

$$\mathcal{L}_{confidence}(\mathbf{\Gamma}|\mathbf{X}) = \sum_{i}^{N_k} \|\delta_i - \hat{\delta}_i\|_2. \tag{6.8}$$

The pose predictions can then easily be obtained by sampling the most confident point correspondences, while removing the initial erroneous predictions right away.

6.1.3 Pose Estimation and Refinement

The initial pose hypothesis, computed from the aforementioned correspondences, is refined as a post-processing step. Following previous works [21], h_p pose hypotheses are sampled, in our case, using N_k number of point correspondences for each. Out of these correspondences, only the 10% points with highest confidence are kept. However, even though we are able to greatly improve the quality of the point correspondences used to compute pose estimates with this step, the initial randomly selected points might still be highly inaccurate, leading to erroneous predictions included in the confidence sampled subset. To overcome this problem, only one hypothesis out of the h_p is chosen, by scoring each hypothesis using the mean confidence over the probabilities of the correspondences used to compute the pose estimate.

Finally, the best hypothesis is refined by repeatedly sampling N_k randomly selected points and re-running the PnP or Kabsch algorithm, including the additional 10% most confident correspondences in this point set. An outline of our algorithm is summarized in Algorithm 3.

Algorithm 3 Confidence-Based Pose Refinement

Input: 2D-3D or 3D-3D point correspondences
Output: estimated camera pose
h_p times randomly sample N_k point correspondences;
$\{\hat{\delta}\}_i \leftarrow \mu_{\mathbf{\Omega}}(\mathbf{p}_{\hat{\mathbf{X}}_w}, \hat{\mathbf{X}}_w)$: estimate confidence for the N_k points
estimate h_p poses on most confident points by solving the PnP/Kabsch;
score $\leftarrow$ mean confidence of point set;
for N iterations **do**
  sample N_k points
  predict their confidence
  refine best hypothesis on most confident points
end for

6.1.4 Implementation Aspects

For scene coordinate regression, a U-Net [178] is deployed as the network architecture, as it could easily be used to regress correspondence maps and has been shown to train well on few input images [178]. Four convolutional layers with pooling layers and dropout applied after each, and four up-sampling layers are used, which gives a final feature map of size $W \times H \times 8$. By applying a last convolutional layer, we obtain the final correspondence map of size $W \times H \times 3$.

For confidence prediction, we deploy a PointNet [172] network architecture. The authors of PointNet, originally deployed for the task of classification and segmentation of a point cloud, designed their network with regard to certain properties aiming to solve the task at hand. Most importantly as a point cloud is an unordered set of points, the prediction of the network should not be dependent on the order of the input cloud. To achieve such permutation-invariance, single multi-layer perceptrons are shared for each of the input points and symmetric functions used for further processing in the network. Second, a point cloud can undergo a rigid transformation and still provide the same geometric structure. Therefore, similar to Spatial Transformer Networks [97], the point cloud is processed by learning a rigid transformation. Finally, global as well as local features provide important information and are therefore extracted to capture the overall structure of the point cloud as well as the relationship between points. We change PointNet with respect to our specific input, $N \times 5$ and output, $N \times 1$, requirements. As our aim is to predict confidences for a subset of point correspondences, we mimic the input for solving the PnP and, keeping our final objective in mind, we feed randomly selected points to the network. This way, the network learns to be independent of the order in which the points are fed and could still be used in scenarios where a dense representation of point correspondences is not given. For the final regression layer, following [234], we first apply a hyperbolic tangent followed by a ReLu activation, so the network can easily predict low confidence for highly inaccurate points. Mostly for evaluation purposes, s (see Equation 6.7) is computed, such that the inlier points have a lower bound probability of 0.75, resulting in $s = 2.8768$.

We set $N_k = 500$ out of which only the most confident points are used to estimate a camera pose, resulting in only $N_{best} = 50$ points. For pose refinement, we follow the parameter settings of [21] and sample $h_p = 256$ initial hypotheses. Then the best one according to its

confidence is refined for eight iterations on the additional most confident points out of the randomly sampled points. To solve the PnP, we use OpenCV's implementation of [122].

Both networks are trained separately for 800 epochs with batch size of 20 using RMSprop Optimizer with an initial learning rate of $5 \cdot 10^{-4}$. All experiments were conducted on a Linux-based system with 64-GB RAM and 8-GB NVIDIA GeForce GTX 1080 graphics card and implemented in TensorFlow [1].

6.2 Experiments

To evaluate our method, we define baseline models, which are described in Section 6.2.2, and use the following metrics. For this purpose, inliers and outliers are defined as $\|\mathbf{X}_w - \hat{\mathbf{X}}_w\|_2 < t_{inlier}$, with $t_{inlier} = 0.1\ m$ being a common threshold chosen to define inliers [21, 199, 220]. Every inlier point is therefore counted as a true positive in our evaluations. For pose estimation we compute the metrics described in Section 2.5 and calculate the median rotation, the translation error and the pose accuracy, where a pose is considered correct if the rotation and translation error are below 5° and $5\ cm$, respectively.

6.2.1 Dataset

Our method is evaluated on the publicly available 7-Scenes dataset from Microsoft, which consists of seven indoor scenes with varying volumes ranging from 1 to 18 m^3. RGB and depth images of each scene and their corresponding ground truth camera poses are available. For each scene images in the range of 1K to 7K are provided, including very difficult frames due to motion blur, reflecting surfaces and repeating structures, for example, in case of the *Stairs* scene. The images were captured using a Kinect RGB-D sensor and the ground truth poses obtained using KinectFusion. Following the state of the art, we use the training and test sets specified for this dataset. No augmentation as proposed in [156, 227] was performed and our models were trained individually for each scene.

6.2.2 Baseline Models

First, we evaluate each individual component of our model and create the baseline models for comparison. To start with, the first step of our pipeline, the scene coordinate regression is evaluated. To this aim, a model is trained on scene coordinate regression by using different loss functions. Mainly we compare between ℓ_1 and the L_{coords} loss as described in section 6.1.1. In this case, a pose estimate is computed by randomly sampling N_k number of points. Results are given for a single pose estimate. Further, we abbreviate these model as P_{ℓ_1} and P_{tukey}. Next, to evaluate the scene coordinate regression quality, regularization is added in the form of the introduced smoothness term L_{smooth}, corresponding to model P_{smooth}.

Further, and more importantly, the second step of our pipeline, the confidence prediction is analyzed. The predicted scene coordinates and associated image points are used to train a model and regress the confidence of each correspondence. For pose estimation, in this case,

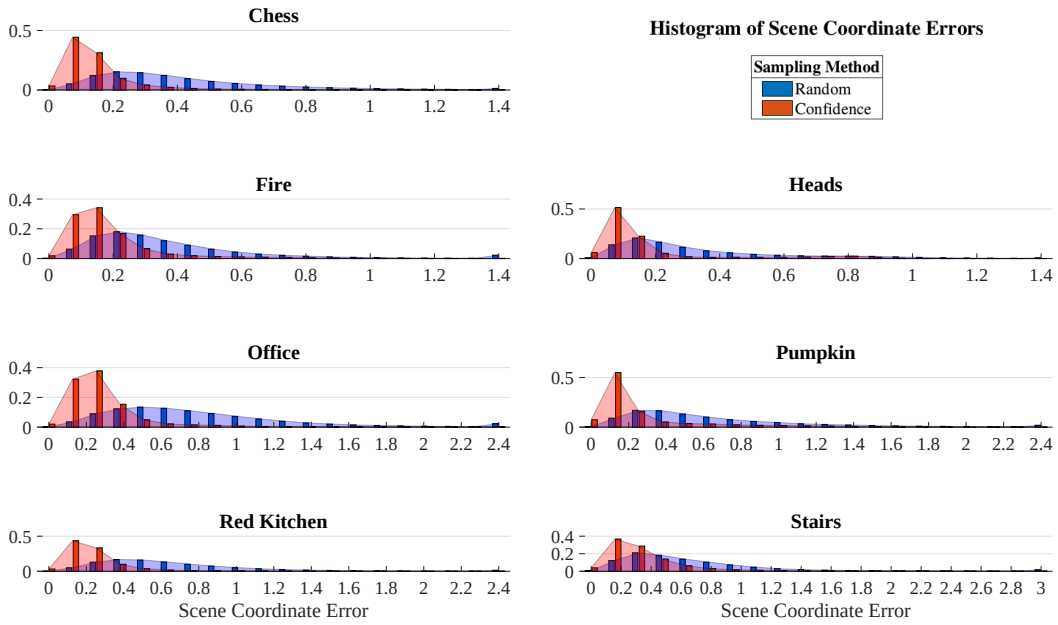


Fig. 6.2. Normalized histograms of the regressed scene coordinate errors on the 7-Scenes dataset. A random subset of points as well as, the most confident points, as indicated by our model, out of the randomly drawn samples is shown. Results indicate that our method can successfully identify outliers and provide confidence values according to the quality of regressed correspondences.

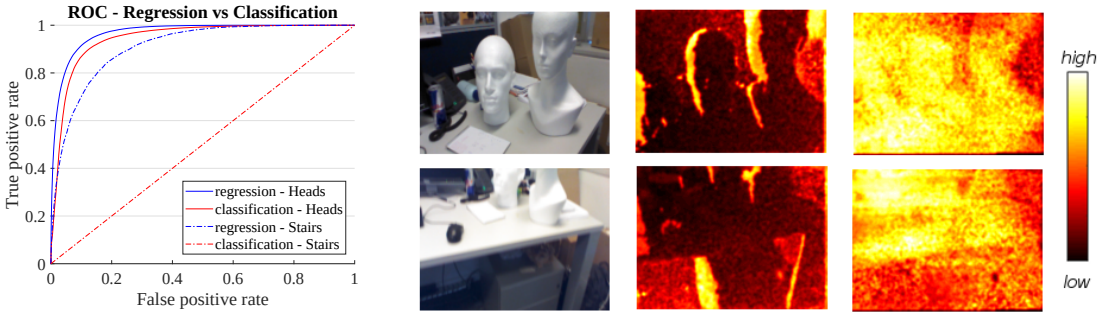
only the most confident correspondences out of the initial randomly sampled N_k points are kept. We abbreviate this model as $P_{confidence}$.

6.2.3 Evaluation of Baseline Models

Each of our models is evaluated, comparing the error between the regressed and ground truth scene coordinates, and the camera pose error. We compare our models trained on the scene coordinate regression and additionally regularizing the model using the smoothness term. Within this evaluation, we found a slight improvement in terms of the regressed coordinates as well as the pose estimates comparing our models with and without additional regularization. Using our proposed confidence prediction, the point errors of the sampled set used to compute a pose estimate significantly decrease, successfully eliminating most of the erroneous predictions and greatly boosting the pose estimation accuracy. Figure 6.2 illustrates our findings. As a result, the estimated poses also improve significantly, as seen in Table 6.1. Specifically, the translation error greatly decreases. It should be noted, that only the most confident point correspondences are used to compute a pose estimate for this model. As a result, more accurate poses are obtained using a much smaller number of points. Additionally, the percentage of inliers in the sampled points used to compute initial pose hypothesis significantly increases. For further evaluation of our proposed confidence prediction, we sampled h_p pose hypotheses and keep the pose hypothesis with the highest inlier score as a final result. Due to the very low amount of inliers, it is difficult to apply RANSAC and obtain satisfactory results without pose refinement or additional confidence prediction. Furthermore, we analyze the quality of our model's coordinate regression and confidence prediction. For this purpose, example error and probability maps are shown in

Tab. 6.1. Median rotation and translation error on the *Heads* scene for our baseline models. Camera pose estimates are computed using Kabsch algorithm for 3D-3D or PnP for 2D-3D correspondences. Results are computed using h_p number of pose hypotheses without any further refinement. In addition, the percentage of inliers for a set of correspondences used to compute a single pose hypothesis is reported.

Model	h_p	P_{ℓ_1}	P_{tukey}	P_{smooth}	$P_{confidence}$
Kabsch	1	$18.0°, 0.29m$	$8.77°, 0.15m$	$6.67°, 0.12m$	$5.77°, 0.07m$
	256	$14.7°, 0.25m$	$6.23°, 0.11m$	$5.88°, 0.09m$	$4.86°, 0.06m$
PnP	1	$50.3°, 0.44m$	$41.3°, 0.44m$	$44.3°, 0.43m$	$10.6°, 0.18m$
	256	$33.7°, 0.46m$	$25.9°, 0.43m$	$26.3°, 0.37m$	$5.09°, 0.10m$
Inliers	1	5.1%	7.8%	11.9%	50.7%



(a) ROC **(b)** RGB image **(c)** Error map **(d)** Confidence map

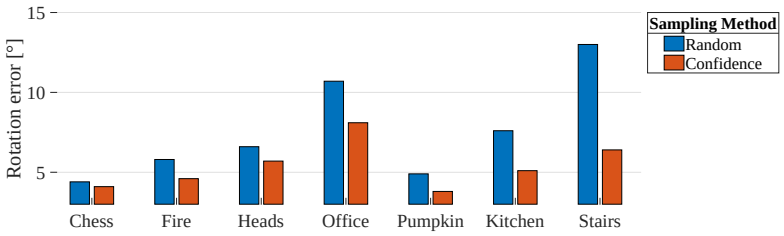
Fig. 6.3. a) ROC comparison between regression and classification for the task of confidence prediction on the training images of the *Heads* and *Stairs* scenes. Example images showing b) an input RGB, c) a color-coded corresponding pixel-wise scene coordinate error map computed with respect to the ground truth and d) a confidence map for each pixel predicted by our model.

Figure 6.3. In our case, since we densely regress the scene coordinates, high error values usually correspond to missing depth values and therefore missing ground truth coordinates in the image. Although it seems difficult for the network to accurately predict low confidences in regions with unusually large error, in regions of inlier predictions, the model is able to predict corresponding high confidences.

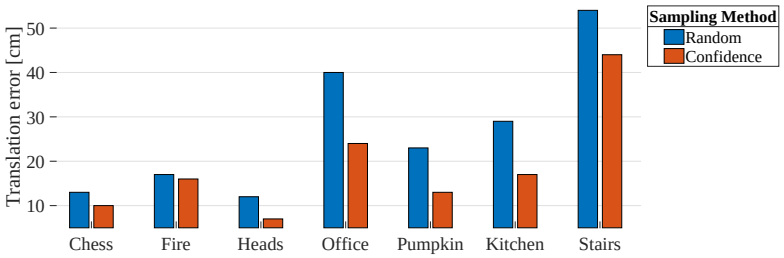
6.2.4 Evaluation of Confidence Prediction

The quality of our proposed confidence prediction is essential for further pose estimation. Therefore, we now first evaluate how to train a confidence estimation network, comparing between a regression and classification approach, and second, evaluate the influence of our confidence based sampling on the resulting accuracy of the estimated camera poses.

Regression versus Classification To assess the quality of regressing correspondence probabilities, a model is trained on simple classification, where a point correspondence could be labeled either as an inlier or an outlier depending on the threshold t_{inlier}. The model is trained using cross-entropy loss. As a second step, we train our proposed model, regressing



(a) Median Rotation Error



(b) MedianTranslation Error

Fig. 6.4. Quality of camera pose estimates with respect to the ground truth when correspondences are randomly sampled in comparison to the proposed confidence-based sampling. Reported are the median rotation and translation errors per scene of the 7-Scenes dataset, showing a clear improvement when using the proposed sampling strategy.

probabilities in the range of $\delta \in [0, 1]$ instead and plot the resulting ROC curves as shown in Figure 6.3a. As a result, we assess that the performance of regression in this case is more or less equal to classification. However, we do not restrict the model to a specific threshold chosen for the inlier definition, which needs to be adapted for each scene depending on the quality of scene coordinate regression. In comparison, a classification model trained on the challenging *Stairs* scene results in a drop of relative rotation and translation error of 4.4% and 31.5%, respectively, because very few inliers were available during training.

Influence on the camera pose accuracy Further, to assess the quality of our confidence prediction, we compute a camera pose estimate from the regressed correspondences. For comparison we either randomly sample a subset of points that is used to compute a pose estimate or use the most confident points indicated by the predictions of our model. Figure 6.4 shows the resulting camera pose errors with respect to the ground truth for both evaluations. In comparison to simply sampling random correspondences a clear improvement can be observed when using the proposed confidence based sampling strategy. This further shows that our network is capable of learning a measure of quality for given point correspondence, estimating high confidence for the most accurate predictions.

Tab. 6.2. Median rotation and translation error on the 7-Scenes dataset in degrees and cm of the state of the art and our method. Percentages are given for poses below 5° and 5cm threshold.

	Method / Scene	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs	Average
RGB Information	LSTM [222]	5.7°, 24cm	11.9°, 34cm	13.7°, 21cm	8.0°, 30cm	7.0°, 33cm	8.8°, 37cm	13.7°, 40cm	9.8°, 31.3cm
	PoseNet [105]	4.8°, 13cm	11.3°, 27cm	13.0°, 17cm	5.5°, 19cm	4.7°, 26cm	5.3°, 23cm	12.4°, 35cm	8.1°, 22.9cm
	VLocNet [219]	1.7°, 3cm	4.9°, **4cm**	5.0°, 5cm	**1.5°**, **3cm**	**1.9°**, **4cm**	**1.7°**, **3cm**	**5.0°**, **7cm**	**3.1°**, **4cm**
	DSAC [21]	**1.2°**, **2cm**	**1.5°**, **4cm**	**2.7°**, **3cm**	1.6°, 4cm	2.0°, 5cm	2.0°, 5cm	33.1°, 117cm	6.3°, 20cm
	Ours2D-3D	1.3°, 3cm	2.9°, 6cm	3.2°, 4cm	2.1°, 6cm	2.9°, **4**cm	2.7°, 5cm	6.3°, 13cm	**3.1**°, 5.8cm
		83.0%	42.4%	59.6%	42.5%	62.2%	58.2%	9.9%	51.1%
RGB-D inf.	Ours3D-3D	1.2°, 3cm	2.7°, 5cm	3.1°, 3cm	2.0°, 5cm	1.3°, 3cm	1.3°, 4cm	6.1°, 13cm	2.5°, 5.2cm
		85.7%	48.8%	**60.1%**	49.2%	66.6%	66.4%	11.6%	55.5%
	SCoRe [199]	**92.6%**	**82.9%**	49.4%	**74.9%**	**73.7%**	**71.8%**	**27.8%**	**67.6%**

6.2.5 Comparison to the State of the Art

Finally, we report the results of our framework using a combination of scene coordinate regression and confidence prediction, described as $P_{confidence}$. We compare our results to the current state-of-the art methods, namely PoseNet [105], which directly regresses the camera poses from the RGB input images and refines the trained models by optimizing on the re-projection error. The median rotation and translation errors evaluated on the 7-Scenes dataset can be found in Table 6.2, where we report the results obtained using PnP ($\text{Ours}_{\text{2D-3D}}$) as well as, given depth information is available, using Kabsch algorithm ($\text{Ours}_{\text{3D-3D}}$). Our model does not depend in any way on the algorithm used to compute pose predictions; therefore, we can easily interchange these algorithms without the need to train additional models.

In most cases, we found a significant improvement in pose accuracy compared to PoseNet [105] and adaptation of this method PoseNet LSTM [222]. A recent method, namely VLocNet [219], has achieved comparable results to our method, however using previous pose information in a direct regression framework. In turn this work should be considered as a tracking scenario instead of a re-localization one.

In addition, we compare to recent works on scene coordinate regression, [21]. With the exception of the challenging *Stairs* scene, the state-of-the art method shows slightly better accuracy in terms of RGB pose estimation considering each scene individually. On average our method shows good performance compared to the state of the art.

Although our confidence prediction significantly improves the results, the initial scene coordinate regression still seems erroneous, which will be further explored in future work considering optimizations in handling missing depth and thus ground truth scene coordinates as well. Given that the depth information is available, improvements of the accuracy using RGB-D information can easily be obtained since neither the models nor the pose refinement rely on these algorithms. The results including RGB-D information can be seen in Table 6.2, for which we give a comparison to the state-of-the art scene coordinate regression forest approach [199]. Further, since we only depend on the most confident points, our results were obtained using a smaller set of points. Further, RANSAC based optimization, as applied in most state-of-the art methods, could be easily applied to obtain more accurate pose estimates.

For evaluation and comparison to current RGB methods, we keep the parameter settings for pose refinement as proposed in [21].

6.3 Remarks and Conclusion

In this work, we present a framework for dense scene coordinate regression in the context of camera re-localization using a single RGB image as the input. When depth information is available for obtaining the camera coordinates, the corresponding scene coordinates could be regressed and used to obtain a camera pose estimate. We incorporate this information into the network and analyze how the scene coordinate regression can be further optimized using a smoothing term in the loss function. In addition and more importantly, we predict confidences for the resulting image point to scene coordinate correspondences, from which the camera pose can be inferred, thus eliminating most of the outliers in advance and greatly improving the accuracy of the estimated camera poses. As a final step, the resulting confidences can be used to refine the initial pose estimates, which further improves the accuracy of our method.

Acquiring reliable correspondences is of high importance for a large number of computer vision applications. With the advent of deep learning numerous works have shown the potential of neural networks in this regard [57]. Most recently, Brachmann et al. [25] incorporate a deep learning based approach into RANSAC-based frameworks. Instead of trying to remove RANSAC, a neural network is trained in such a way that good correspondences are more likely to be chosen for hypotheses prediction. By including this approach into their previous work for camera localization [24], they show significant improvements for the task at hand, which underlines the importance and potential of deep learning solutions to handle erroneous predictions and outliers. Uncertainty estimation within this context is yet to be explored by the computer vision community and would pose another interesting research direction for future work.

Although, in general providing good localization performance, structure-based methods in one form or the other rely on a reconstruction of the scene as prior information and if not available or inaccurate can lead to poor performance. Building an accurate reconstruction becomes more challenging with increasing scene size, which for instance is of particular interest in autonomous driving scenarios, that naturally require maps of large scale. In the next chapter, we therefore analyze methods solely relying on 2D image information. In particular we aim to assess methods that directly regress the camera pose of a corresponding RGB image.

Part IV

Direct Pose Regression

7 Introduction

7.1 Motivation

RGB cameras have become available on most systems as well as mobile devices and thus provide important visual information at a low cost. Therefore, a lot of recent research has focused on solving computer vision tasks such as object pose estimation or camera localization, solely relying on RGB information.

In this context, very recently (in 2015), direct camera pose regression approaches have started to emerge, mainly using convolutional neural networks to estimate the rotation, most often represented as quaternions, and position of the camera given a single RGB image as input. With only one forward pass of the neural network, these methods can be applied in real-time applications and due to low memory footprint can easily be deployed on mobile devices as well.

However, without the additional 3D information in comparison to for example structure-based methods, systems have to become more robust to handle challenges such as varying illumination conditions or occlusions.

In outdoor environments, especially in autonomous driving scenarios, where mounted RGB cameras can be part of the standard equipment, visual cues are used to detect important information such as traffic signs and process the environment for a secure movement of the vehicle. Additionally the information can be used for large-scale navigation, especially when other sources of information, such as GPS, fail. Nevertheless, long-term and large scale localization has posed a challenging task. Highly dynamic environments, created by moving vehicles/objects and pedestrians, as well as strong weather and seasonal changes such as rain, snow, daylight and night conditions can make localization from RGB images unreliable.

Considering indoor environments, direct regression methods have shown to lack generalization capabilities. The need for training a scene-specific model first limits their applicability for online applications such as SLAM. Further currently these methods lack the general accuracy in comparison to structure-based methods. In their defense, however, structure-based methods rely on computationally rather expensive outlier handling such as RANSAC and camera pose refinement strategies, that are essential to provide accurate and robust solutions. Furthermore, due to visual similarities, direct regression methods have difficulty in handling ambiguity arising in the scene due to symmetries and repetitive structures. Therefore, images taken at highly different viewpoints can result in visually similar appearances and as such easily confuse methods that solely rely on RGB information.

In this thesis, in the latter, we mainly attempt to address two aspects that occur in direct camera pose regression methods to improve said methods while retaining the advantages they have to offer. First, we attempt to refine a camera pose estimate of a direct regression method, however, still solely relying on RGB information. Due to the lack of geometric information available this poses a highly challenging task. Further, the computational burden of any refinement process has to be kept in mind such that the advantage of fast deployment of direct regression methods can be maintained. And second, we propose a framework specifically designed to handle ambiguities arising in the scene while maintaining the original accuracy with minimal computational overhead in non-ambiguous environments.

7.2 Related Work

Before describing the work conducted in the scope of this thesis, we start with a general introduction of direct camera pose regression methods with neural networks that have recently been proposed and published at high value computer vision conferences.

7.2.1 Absolute Pose Estimation

Starting with the introduction of PoseNet [107], Kendall et al. presented a computationally very fast solution for solving the camera pose estimation problem relying solely on RGB information and also showing great capabilities when being applied on large-scale scenes. Given an RGB input image, the methods uses a convolutional neural network to directly regress the camera pose with one forward pass of the network. This, on the other hand, came at a large drop in general accuracy compared to earlier state-of-the-art methods. Thus, several extensions and modifications of this method have been proposed. In a subsequent work Kendall et al. [105] propose the use of a novel loss functions for direct camera pose regression, which includes learning the homoscedastic task uncertainty. Further, in case depth information is available during training, the authors additionally apply a loss based on minimizing the re-projection error between projections of the 3D points when using the ground truth and predicted camera poses.

Relative and temporal constraints to improve global localizaton have been enforced in a line of works [26, 50, 186, 222, 229]. For instance, Walch et al. [222] use Long-Short-Term-Memory (LSTM) units to reduce the dimensionality of the learned feature representation of the neural network for improved localization accuracy and to alleviate over-fitting. Usually applied to temporal data LSTM units have been shown to perform well on task such as language modeling [209] and understanding [233] or action recognition [221]. In [222] the authors instead apply four LSTM units to find correlations in image representations and in turn learn more suitable features for direct pose regression.
VidLoc [50] uses recurrent neural networks for localization in image sequences. Assuming temporal information is available, LSTM units are used to leverage relevant information, such as movement between frames, for camera pose estimation. In addition mixture density networks are used to regress distributions over the pose space and obtain a measure of uncertainty in the network's prediction.
AnchorNet [186] defines anchor points and proposes offset regression relative to these anchor

points for improved localization. Intuitively based on how humans localize themselves, anchor points are defined in intervals over the map and the best suited anchor point, as well as its offset to it, is predicted by a neural network. This way, performance can be improved in comparison to directly regression all 6 DoF of the camera pose.
MapNet [26] proposes to include relative pose constraints to guide direct camera pose regression, such that the relative pose between predicted camera poses are forced to correspond to the ground truth relative pose or visual odometry of associated image pairs. They further show how to incorporate prior information, if available, such as GPS coordinates. Finally, the resulting camera poses are optimized by applying pose graph optimization. In this case absolute poses, nodes of the graph, are refined such that they comply with the constraints of the relative poses between nodes.

Recently Huang et al. [94] extend pose regression methods to appropriately handle dynamic environments, such as pedestrians or moving objects, that should not affect the final camera pose. An estimated segmentation mask is used to provide probabilities for such objects, that during training as well as testing can be used to apply dropout at pixels and remove such misleading information. Additionally a self-attention module is used to estimate weights, that, in turn weigh the influence of each feature map to the final camera pose regression layers. Finally, multiple hypothesis are sampled using Dropout and the camera pose estimates refined with a uncertainty-aware pose graph optimization.

Another line of research includes uncertainty estimation [106] approximating Bayesian Inference by utilizing Dropout in neural networks, utilizing mixture density networks to predict uncertainty as the variance of the predicted distribution [50], or combining learned feature representations and Gaussian process regressors [38]. However this particular research direction will be in more detail addressed in Chapter 9. Although there has been a vast amount of research addressing direct camera pose regression, the accuracy gap and lack in generalization capabilities of the aforementioned methods in comparison to their geometric counterparts remains significant.

Sattler et al. [193] therefore investigate the limitations of direct camera pose regression methods and their connection to the underlying network architecture and shows that current architectures mostly learn a conditional average and interpolate between the poses seen during training, therefore strongly resembling a nearest neighbor estimation of the pose.

7.2.2 Relative Pose Estimation

However, inferring the relative pose between two frames has shown to be less affected by the problems absolute camera pose regression methods suffer from. Therefore a line of research has aimed to use relative pose information between pairs of frames for accurate absolute pose prediction. The basis of these methods is a two stage approach, first retrieving the nearest neighbor of a database associated with absolute poses. Then, as a second stage the predicted relative pose between the database and the query image is used to predict the absolute pose for the query image. The general framework of such methods is depicted in Figure 7.1 For robust prediction, in most cases, k nearest neighbors are used instead of a single one and the query pose computed by triangulation [6, 144].

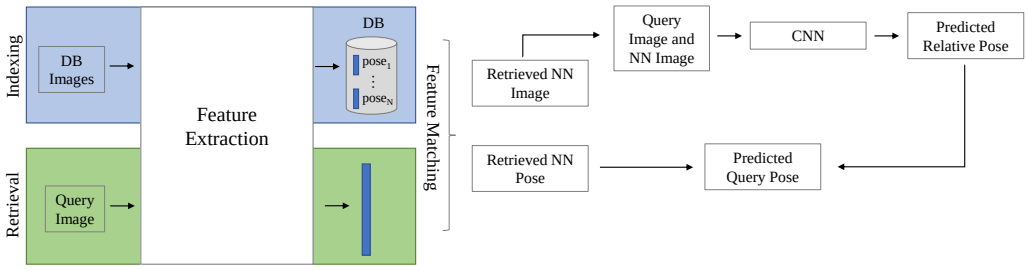


Fig. 7.1. General pipeline for relative pose estimation methods. Features are extracted for the training images to form a database of features and associated pose labels. For a given query image the nearest neighbor in the feature space is retrieved and with the nearest neighbor image and the query fed to a neural network that is trained to predict the relative camera motion between two frames. The nearest neighbor pose is then updated with the relative motion resulting in the estimated camera pose of the query.

Melekhov et al. [144] show that learned feature representations using convolutional neural networks can outperform traditional ones, like SIFT, for nearest neighbor retrieval. Later, the authors show the benefits of relative pose regression in generalization to unknown scenes.

In [6] the authors of RelocNet propose a novel similarity measure for training localization specific features based on the volume of the camera frustum overlap between to cameras such that image pairs with high overlap are close in feature space. Relative poses are regressed directly in $SE(3)$ and in addition the method shows to some extend generalization to novel scenes that have not been used during training.

With CamNet [58], Ding et al. propose a coarse to fine learning strategy, training several branches of a neural network to predict either coarse or fine relative poses. For a query image the camera pose can then be obtained by applying the relative motion in stages, each stage providing a more accurate solution.

Instead of predicting the relative pose, NC-EssNet [246] proposes to estimate the essential matrix and retrieve the relative pose from the essential to alleviate the problems of balancing rotation and translation in direct regression methods. The authors further highlight the problems of current regression methods in comparison to traditional methods, for example using a simple SIFT feature matching and essential matrix computation from obtained 2D-2D image matches using RANSAC.

In the next chapters we aim at addressing some of the problems arising in direct camera pose regression methods. First, as RANSAC is a rather computationally expensive procedure, we investigate a deep learning based solution for camera pose refinement that solely relies on RGB information.

8 Learning Pose Refinement

8.1 Motivation

Direct camera pose regression methods have shown to provide computationally efficient alternatives for visual localization, while solely relying on RGB information to provide an initial pose estimate. However, in comparison to their structure-based counterparts, a significant gap in general accuracy can be observed. Due to the lack of geometric information and sufficient training sets available these methods often do not generalize well to novel viewpoints. Considering structure-based methods, however, initial pose estimate are no more accurate than those of direct regression methods, but can be refined using the 3D information of the corresponding scene. With only color information considered, direct regression methods do not offer the same refinement possibilities and thus such methods have not yet been proposed in the current literature. With the recent advances of deep learning, however, the possibility of simple RGB refinement has become an actual alternative. Therefore, in this chapter, we aim to explore a deep learning alternative for the task of camera pose refinement. While solely relying on RGB input images, the idea is to capture the geometric information between a camera pose and the corresponding RGB images with a neural network, and implicitly encode such information. For this aim, we are inspired by the concept of generative adversarial networks, that allow us to model the distribution, in our case the joint distribution of camera poses and input images.

8.2 Background

Before diving into the details of our method, we briefly describe some of the major refinement strategies used for object as well as camera pose refinement in computer vision applications. A general approach, mainly used for registration but not limited to it and applied in various other applications such as object pose estimation [104] or reconstrucion [159] and operating on points clouds, is the iterative closes point (ICP). In the context of camera localization this entails an adaption of RANSAC, which is currently commonly used in structure-based methods. Further, we briefly describe the concept of GANs that in large parts inspired our proposed framework.

8.2.1 Camera Pose Refinement

We start by summarizing two main strategies used for camera pose refinement, the iterative closest point algorithm and RANSAC-based refinement.

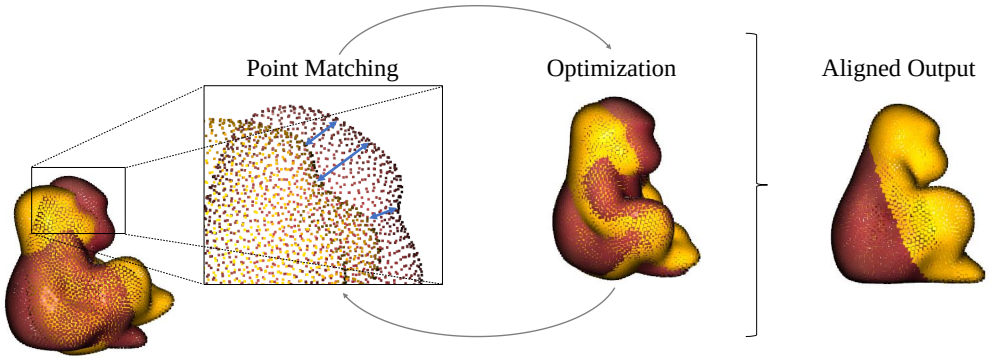


Fig. 8.1. Schematic overview over the ICP algorithm. The alignment of two models is computed by estimating the transformation between them in an iterative fashion. Each iteration consists of two steps, point matching (based on their distance) and transformation optimization.

Iterative Closest Point

If a 3D point cloud is available, iterative closest point [10] can be computed between a model and for instance the point cloud or partial estimation of it transformed by a predicted pose. The algorithm iteratively tries to align the point clouds by matching a point to its closest point in the reference and minimizing the alignment error between the matches. A schematic overview is depicted in Figure 8.1 Starting from an initial alignment of two points clouds an optimal transformation is computed that minimizes a cost function, e.g. the distance between the known points of the known model and the transformed corresponding points. As a result one obtains the transformation that best aligns the two points clouds, preferably obtaining a more precise pose estimate as well. Variations of the method have been proposed throughout the years [183, 231], addressing for instance which points to use for the computation [141, 224], how to match points between the two clouds, weighting [78] and rejection of point matches [141] and the choice of the objective function. For example Chen and Medioni [45] propose a point-to-plane instead of point-to-point distance. Another prominent example that uses point-to-plane distance metric and ICP for alignment of a RGB-D frame to the model is KinectFusion [159], that as a first method, computes a dense reconstruction from a hand-held device and RGB-D videos in real-time.

RANSAC-based Refinement

Due to large amount of outliers or noise in the data, most state-of-the art localization methods currently employ a version of RANSAC or ICP to obtain a robust solution to the camera pose estimation problem. The performance of RANSAC, however, strongly depends on the number of outliers in the data, which in turn correlates with the computational time required to guarantee an accurate solution. In addition, these methods most often rely on 3D information to perform 2D-3D matching. Subsequent pose refinement is then applied by recomputing the camera pose on the enlarged set of inliers found and iteratively computed until a certain criteria is met. An outline of such an algorithm can be found in Chapter 2.

8.2.2 Generative Adversarial Networks

Introduced by Goodfellow et al., Generative Adversarial Networks (GANs) [79] have recently shown great success in generating images as well as improving the performance of deep neural networks trained for tasks such as object detection [223], human pose estimation [46, 232] or realistic image composition [126]. Such GANs consist of two networks, a generator G that captures the underlying data distribution and a discriminator D. The generator tries to reconstruct a sample $\mathbf{X}$ from the latent variable $\mathbf{z}$ and the discriminator tries to estimate the probability of a sample coming from the actual distribution $D(\mathbf{X})$ or the generated one $D(G(\mathbf{z}))$, i.e. can tell the real distribution and distribution of generated data apart. During training, the two networks are in competition with each other as the generator tries to better mimic the ground truth data distribution such that it becomes more and more difficult for the discriminator to correctly classify a sample representation. More precisely, in every training step, the generator is updated in a way such that it is more likely to fool the discriminator. This results in a min-max game with the following objective function:

$$\min_G \max_D \mathcal{L}(D,G) = E_{\mathbf{X}\sim p_{data}(\mathbf{X})}[\log D(\mathbf{X})] + E_{z\sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))]. \tag{8.1}$$

Conditional GANs A conditional version of GANs was proposed in [147], additional feeding prior information to the GAN framework. Depending on the application, the condition can vary, such as class labels, so that the generator learns to reconstruct an image of a type of object for instance. The condition is fed to both the Generator and the discriminator as additional information, such that given a condition y the objective function becomes

$$\min_G \max_D \mathcal{L}(D,G) = E_{\mathbf{X}\sim p_{data}(\mathbf{X})}[\log D(\mathbf{X}|y)] + E_{z\sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z}|y)))]. \tag{8.2}$$

Conditional GAN have successfully been used in various applications such as image-to-image translation [96], domain adaptation [218] or segmentation [205].

8.3 Related Work

Since in this work, we want to explore re-localization methods utilizing RGB information only, we build on top of recent research on direct camera pose regression methods. In that sense, direct camera pose regression methods such as PoseNet [107] and related works are strongly relevant for ours. However, we additionally attempt to model the connection between an RGB image and its camera pose implicitly, rather than trying to simply learn this mapping directly. For this purpose, we show the advantage that leveraging an adversarial network can have on such methods. Therefore, we propose a framework based on a camera pose regression network and a discriminator network that, given a regressed pose conditioned on the input image, learns to distinguish between regressed and ground truth poses. Further, once the model has learned a representation of this connection, as our main contribution, we show how the trained model can be used for camera pose refinement. By leveraging the learned information encoded in the discriminator network, the localization accuracy can be improved beyond the one of a simple camera pose regression network.

Generative Adversarial Networks [79], allow to model both unconditional and conditional data distributions and have recently shown great success in improving the performance of deep neural networks trained for tasks such as object detection [223], human pose estimation [46, 232] or realistic image composition [126]. However, to the best of our knowledge GANs have not yet been explored in the context of camera pose estimation.

There have been several methods addressing pose refinement in the context of an object's as well as a camera's pose with neural networks. For instance, Manhardt et al. [136] and Li et al. [125] propose a neural network that given a pose estimate used the CAD model of an object to render that object from the predicted viewpoint and refine the pose by minimizing the error between rendering and observed view in an iterative fashion. DSAC [21, 24] in comparison relies on RANSAC and iterative refinement on inlier point correspondences to obtain an accurate camera pose prediction.

The above methods, however, rely on additional information in the form of a model or depth information or produce rather computationally expensive refinement strategies.

8.4 Methodology

Our methodology is inspired by GANs in the sense that we are aiming to train a discriminator network that tries to predict whether a pose estimate and the corresponding input image are part of the distribution seen during training or significantly varies from said distribution. The work presented in the following is part of the published paper

'Adversarial Networks for Camera Pose Regression and Refinement' by Mai Bui, Christoph Baur, Nassir Navab, Slobodan Ilic and Shadi Albarqouni, published in Proceedings of the International Conference of Computer Vision (ICCV), 2nd Workshop on Deep Learning for Visual SLAM, 2019, Copyright 2019 IEEE [33].

As introduced by previous camera pose regression approaches, our baseline model consists of a convolutional neural network, parameterized by $\mathbf{\Gamma}$. We hereby referred to this network as the pose regressor, which learns the mapping $\mu_{\mathbf{\Gamma}}(\mathbf{X}) : \mathbb{R}^{W \times H \times 3} \mapsto \mathbb{R}^7$ between an input RGB image $\mathbf{X} \in \mathbb{R}^{W \times H \times 3}$ and an absolute camera pose $\mathbf{p} \in \mathbb{R}^7$.

Additionally we attempt to learn the distribution of camera poses and their respective RGB images captured by the camera. More precisely, we aim to train a pose discriminator network that should be able to distinguish between regressed and ground truth camera poses with respect to the corresponding input image. Following the general GAN framework, the pose regressor and discriminator are trained in an alternating manner, where the pose regressors goal is to fool the discriminator, such that it can not clearly distinguish between regressed and real camera poses anymore. Finally, once the discriminator has learned the joint distribution of camera poses and input images, that is, has modeled the geometric mapping between an input image and a camera pose, we leverage the information captured by the discriminator to update and refine the regressed camera pose. By freezing the discriminator networks weights and optimizing solely the regressed camera pose according to the computed gradient

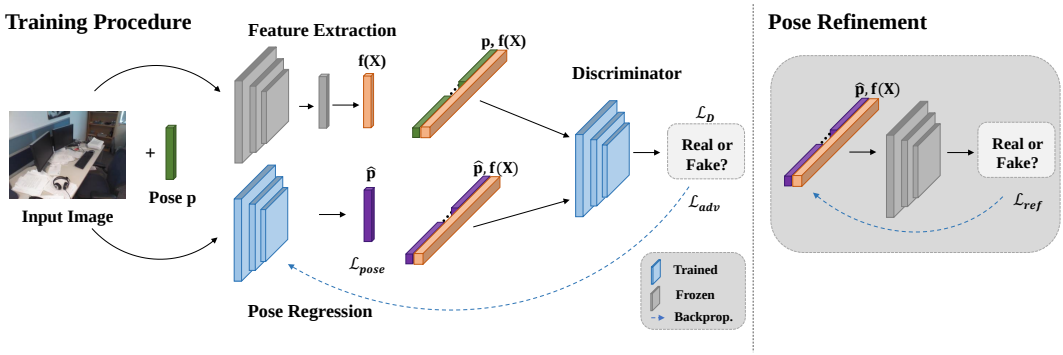


Fig. 8.2. Given an RGB image, a corresponding camera pose is estimated with a pose regression network. Alongside the estimated pose, a feature representation of the corresponding image is extracted and used to train a discriminator network. This network is trained to distinguish between ground truth and regressed poses considering the input image and can then be leveraged to refine the regressed camera pose.

information, we aim at pushing the regressed pose closer towards the manifold of real poses to ultimately better fit the input image and preferable obtain a more accurate camera pose. An overview of our method is depicted in Figure 8.2.

Camera Pose Regression Given an RGB image $\mathbf{X} \in \mathbb{R}^{\mathrm{W}\times\mathrm{H}\times 3}$, with W and H being the image width and height respectively, our objective is to predict the camera pose $\mathbf{p} = [\mathbf{q}, \mathbf{t}]$ given as orientation, represented as vector $\mathbf{q}$, and translation $\mathbf{t} \in \mathbb{R}^3$. For this aim, a convolutional neural network is trained on the homoscedastic loss function

$$\mathcal{L}_{pose}(\mathbf{\Gamma}|\mathbf{X}, \mathbf{p}) = \|\mathbf{t} - \hat{\mathbf{t}}\|e^{-\beta} + \beta + \|\mathbf{q} - \hat{\mathbf{q}}\|e^{-\alpha} + \alpha, \tag{8.3}$$

where $\hat{\mathbf{t}}$ and $\hat{\mathbf{q}}$ represent the predicted translation and rotation, respectively. The parameters β and α are trainable and used to balance both distances, and $\|\cdot\|$ is chosen to be the ℓ_1 norm. Readers are referred to [105] for further details about the loss function and its derivation.

The parameterization used to regress the rotational component of an object or a camera pose with neural networks has been extensively addressed in many recent literature [26, 60]. In this work, we experiment with two representations. First, we choose to evaluate our method on the representation of quaternions, which is already well established in image-based localization method. In this case, a quaternion can be described as $\mathbf{q} = [w, \mathbf{u}] \in \mathbb{R}^4$ where w is a real valued scalar and $\mathbf{u} \in \mathbb{R}^3$. Normalization during the training has to be applied, to ensure that the resulting quaternions lie on the unit sphere. The use of ℓ_1 or ℓ_2 norm in the loss function results in an approximation of the true distance between two quaternions on the unit sphere. However, as mentioned in [105], the resulting quaternions become sufficiently close to the ground truth such that there is no significant difference in ℓ_1 norm and spherical distance, therefore no additional constraints have to be enforced while training the pose

regression network. As a second representation, we use the logarithm of a unit quaternion, or axis angle representation, which is computed as

$$\mathbf{q}_{log} = \log \mathbf{q} = \begin{cases} \frac{\mathbf{u}}{\|\mathbf{u}\|} \arccos(w), & \text{if } \|\mathbf{u}\| \neq 0 \\ \mathbf{0}, & \text{otherwise} \end{cases}, \tag{8.4}$$

and has the advantages of not being over-parameterized, reducing the number of parameters to the minimum of three. Further, it relaxes the need of normalization during the training. The the unit quaternion can be recovered by $\mathbf{q} = [\cos(\|\mathbf{q}_{log}\|), \frac{\mathbf{q}_{log}}{\|\mathbf{q}_{log}\|} sin(\|\mathbf{q}_{log}\|)]$.

Discriminator Both the regressed poses $\hat{\mathbf{p}}$, and the ground-truth poses $\mathbf{p}$, and a lower dimensional representation, $f(\mathbf{X})$, of the corresponding input images, form "fake" and "real" examples, respectively. These pairs of samples are then used to train the discriminator network. The aim of this network is to minimize the following loss function defined as

$$\mathcal{L}_D(\mathbf{\Gamma}|\mathbf{X}, \mathbf{p}) = \sigma(\{f(\mathbf{X}), \mathbf{p}\}, C_{real}) + \sigma(\{f(\mathbf{X}), \hat{\mathbf{p}}\}, C_{fake}), \tag{8.5}$$

where $\sigma(\cdot, \cdot)$ is the binary cross-entropy loss, C_{real} and C_{fake} are set to 1 and 0, respectively. Therefore, the discriminator models the conditional distribution $P(y|\ \mathbf{p}, f(\mathbf{X}))$ of $y \in \{C_{real}, C_{fake}\}$ conditioned on the pose $\mathbf{p}$ and image features $f(\mathbf{X})$, and thus implicitly captures the joint distribution of $\mathbf{p}$ and $\mathbf{X}$. Our framework is, in fact, inspired by GANs to ensure that the geometric mapping between camera poses and the corresponding RGB images are exploited in the network. However it differs from the original GAN framework as our pose regression network is purely discriminative. Providing a feature representation of the image in combination with the pose has shown to be crucial in our experiments, see 8.5.3. Based on intuition this is a result of providing the network with the necessary visual information in combination with the pose, that is with the mapping we are aiming to learn.

Feature Extraction A pre-trained network architecture on ImageNet [184], see Section 8.5.3, is used to extract a feature representation $f(\mathbf{X})$ given an RGB input image. The weights of the network are frozen during the training, as its purpose is mainly to provide the discriminator with a lower dimensional representation of the image. Given the fact that most of the state-of-the-art network architectures produce a rather high dimensional feature representation (compared to the six or seven dimensional camera pose vector), we apply a linear mapping to better balance the dimensionality between feature representation and camera pose. For this aim, we draw inspiration from the concept of dimensionality reduction. To easily integrate the linear mapping into the network architecture, we simply add one additional fully-connected layer, without bias or activation function, right after the trained feature representation layer, and keep its weights frozen during training. This way, the discriminator is discouraged to adapt the extracted features during training and solely base its decision on the features themselves. More sophisticated dimensionality reduction techniques could be applied instead. We leave this as future work. The camera pose vector is then copied, to fit the dimensionality of the extracted feature representation, and concatenated with said representation to form a feature map that is used as the input to the discriminator network. Intuitively we would want the discriminator to learn the connection between RGB images and corresponding camera poses. Therefore, such that the network is discouraged to solely focus on the information provided by either one, the design choices described above were made. However, in addition we have

experimented with fine-tuning the feature extraction network as well as only fine-tuning individual layers. Both resulted in worse performance.

Adversarial Learning Following the training procedure introduced for generative adversarial networks, we alternate between training the camera pose regressor and the discriminator network, updating the regressor on

$$\mathcal{L}_G(\mathbf{\Gamma}|\mathbf{X},\mathbf{p}) = \mathcal{L}_{pose}(\mathbf{\Gamma}|\mathbf{X},\mathbf{p}) + \lambda \underbrace{\sigma(\{f(\mathbf{X}),\hat{\mathbf{p}}\},C_{real})}_{\mathcal{L}_{adv}(\mathbf{\Gamma}|\mathbf{X},\mathbf{p})}, \tag{8.6}$$

such that the network learns to predict reasonable poses with regard to the ground truth as well as more and more realistic poses that should eventually be able to fool the discriminator. Here, the parameter λ balances the influence of the adversarial loss on the pose regressor. Therefore, the adversarial loss acts as an additional regularization to the pose regression network.

Pose Refinement At the end of training the model should be converged and the discriminator successfully "fooled", meaning it can not distinguish properly between regressed and ground truth poses with respect to the input image. Once trained the discriminator network can be used during testing to refine the regressed camera poses. For this aim, the test image is fed to the pose regression network as well as the feature extractor to obtain an initial pose estimate and feature representation. Then, the predicted pose together with the extracted feature representation of the image is used as input to the discriminator, same as during the training stage. However, in succession, the weights of the discriminator are frozen, and the initially regressed pose $\hat{\mathbf{p}}$ for the image $\mathbf{X}$ is updated in an iterative fashion by optimizing the pose parameters over the following loss function:

$$\mathcal{L}_{ref}(\mathbf{\Gamma}|\mathbf{X},\mathbf{p}) = \sigma(\{f(\mathbf{X}),\hat{\mathbf{p}}\},c), \tag{8.7}$$

where the class label c is set to 0.5. This stems from the fact, that at the end of training, the discriminator will not be able to distinguish between regressed and ground truth camera pose anymore, thus predicting values close to 0.5 in both cases, which we have found to be true by experimental evaluation. Intuitively, this amounts to moving along the manifold towards a region where the discriminator reliably confuses real and regresses poses with respect to the input image. Therefore, any predicted pose of an unseen query image should be pushed towards this manifold. As the gradients coming from the discriminator do not necessarily follow a geometrically meaningful direction, we add an additional constraint in the update of the pose vector. Therefore, in case of using the quaternion representation, we restrict the quaternion update, such that its movement along the unit sphere is enforced [14, 37]. This results in an update for one iteration of the following form

$$\mathbf{q}_t = \mathbf{q}_{t-1}\cos(\gamma l) + \frac{\mathbf{v}}{\gamma}\sin(\gamma l), \tag{8.8}$$

with $\gamma = \|\mathbf{v}\|_2$, l being the step size, and $\mathbf{v} \in \mathbb{R}^4$ being the projection of the quaternion gradient $\nabla\mathbf{q}$ into the tangent space. The projection is given as

$$\mathbf{v} = (\mathbf{I}_{4\times4} - \nabla\mathbf{q}\nabla\mathbf{q}^T)\nabla\mathbf{q}, \tag{8.9}$$

where $\mathbf{I}_{4\times4}$ is the identity matrix. To further ensure that the resulting poses are valid, the updated quaternion is normalized after each iteration. However, no such constraints have to be or can be enforced to update the translational component of the camera pose. Though, for simplicity, it is updated with the same step size l.

Training Procedure and Implementation Details There are some aspects to be considered when training generative adversarial networks. Therefore, we now briefly describe our training procedure as well as give implementation details, before moving on to the analysis and experimental evaluation of our method.

Training Stage Stable training and convergence has shown to be an issue in training generative adversarial networks, which we have found to be true in our case as well. Therefore we carefully adjust our training procedure to obtain a good performance of the model. As a first step, the pose regression network is trained for a few epochs to initially give reasonable poses, such that the discriminator can actually be provided with meaningsul information. We then include the adversarial loss in the training procedure and train the discriminator. The parameters β and α are set following the state-of-the-art method of Brahmbhatt et al. [26] and λ is set to $1 \cdot 10^{-3}$. Afterwards, the pose regressor and discriminator are alternately trained on the $\mathcal{L}_G(\mathbf{\Gamma}|\mathbf{X}, \mathbf{p})$ and $\mathcal{L}_D(\mathbf{\Gamma}|\mathbf{X}, \mathbf{p})$ loss functions, respectively, such that regressor and discriminator are in competition with one another as originally described in the GAN framework.

Instead of injecting random noise into the pose regressor, as originally required for the conditional GAN framework, we also experimented with placing dropout layers before every convolutional layer and kept them active both during training and testing. This strategy has already been employed for Image-to-Image translation [96], where it has been pointed out that explicit noise would simply be ignored by the model. However, we have not found significant changes in either training nor the overall performance of the method by employing this strategy.

Implementation Following the state of the art [26], input RGB images are resized to a resolution of 256 pixels in height, normalized, and then used to form mini batches of size 64 to train the neural networks. As the base network of the camera pose regressor a ResNet-34 network architecture is employed, where the final classification layer is removed and instead two fully connected layers for camera pose regression are added after the average pooling layer. The discriminator is formed by a simple network architecture consisting of three convolutional layers, where exponential linear units are deployed as activation functions. All networks are implemented in PyTorch [163]. For Optimization we chose Adam Optimizer with a learning rate of $1 \cdot 10^{-4}$ and train our models for 300 epochs on an 11GB NVIDIA GeForce RTX 2080 graphics card. Once the networks are trained, the regressed camera poses are refined as described in Section 8.4 until convergence, but up to a maximum of 50 iterations at a step

size of $l = 1 \cdot 10^{-3}$. An analysis of the effect of the step size and the number of iterations on the resulting pose accuracy can also be found in more detail in Section 8.5.2.

8.5 Experiments and Evaluation

We evaluate our method on the publicly available 7-Scenes [199] dataset, which is widely used for analysis of camera localization methods and which we have already experimented with in Chapter 6. To briefly summarize, the dataset consists of RGB-D frames of seven indoor scenes, captured with a hand-held Kinect sensor, and corresponding ground truth camera poses computed using Kinect Fusion. The scenes are of varying spatial extent and also differ significantly in the amount of training data available for each scene. Training and test data are specified and consist of distinct camera trajectories. It has been widely used to evaluate camera re-localization methods as it contains several challenging scenarios such as motion blur, repeating structures and texture-less surfaces.

For evaluation, we utilize the base model of the recent state-of-the-art method and implementation of MapNet [26]. We focus on directly regressing the camera pose without the aid of temporal or geometric information as introduced as part of the method's contribution. We investigate the effect of our method on models regressing quaternions themselves, which would result in a PoseNet [107] model where the base network is exchanged with a ResNet. Further, we evaluate the logarithm of a quaternion, therefore comparing to the baseline models of [26].

To summarize, for evaluation of our framework, we introduce the following baseline models and naming conventions:

- **Baseline:** As a baseline model, we train the camera pose regression network on the $\mathcal{L}_{pose}$ loss, which, as already mentioned, effectively results in the state-of-the art baseline method of [26]. However, we abbreviate this model as *Base Model* whenever experiments are conducted by us to explicitly highlight re-trained models and to better analyze the effect of our contributions.

- **Adversarial Pose Regression:** To analyze the effect of adversarial training on the camera pose regression, the regression model is trained on the $\mathcal{L}_G$ loss function (Eq.8.6), which includes a pose regression loss $\mathcal{L}_{pose}$ as well as an adversarial loss term $\mathcal{L}_{adv}$, abbreviated as *Ours*.

- **Pose Refinement:** Finally, during testing, the trained discriminator network is used to further improve the regressed poses using the $\mathcal{L}_{ref}$ loss. The models are then abbreviated as *Ours+Ref*.

For the remainder of this section, these models will be used to validate our method. We start by investigating the effect of optimizing a camera pose regression network including the adversarial loss, after which we quantitatively as well as qualitatively analyze the effect of the proposed pose refinement on the localization accuracy. Finally, setting our method in the

Tab. 8.1. Effect of adversarial training and pose refinement on the camera pose accuracy, evaluated on the *Heads* scene. Median rotation and translation errors are reported. Optimizing the camera pose regression network with the adversarial loss results in an improvement in accuracy, which is further increased by our proposed camera pose refinement.

Scene		Base Model	Ours	Ours+Ref.
Heads	Rotation	14.5°	14.1°	12.4°
	Translation	0.18m	0.17m	0.16m

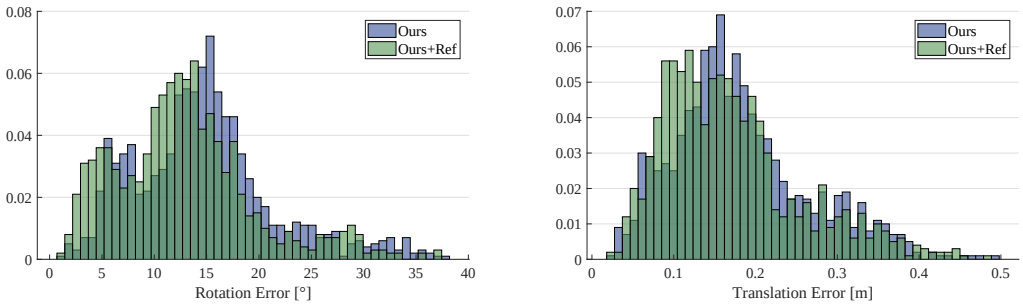


Fig. 8.3. Normalized histograms of rotation and translation errors before and after pose refinement on the *Heads* scene. Results without refinement (Ours) are shown in blue, whereas errors after refinement (Ours+Ref.) are displayed in green.

context of recent research, we compare our results to the current state-of-the-art methods on direct camera pose regression.

8.5.1 Adversarial Learning

First, to investigate the effect of adversarial learning on the camera pose regression framework, we compare rotation and translation errors of our baseline, *Base Model*, and the model *Ours*. The results can be seen in Table 8.1, showing median rotation and translation errors of the described models on the *Heads* scene. That adversarial training can help in training deep networks has already been shown, for example in [232] for the task of human pose estimation, which, however differs significantly from the task of predicting the camera pose from a corresponding image. Nevertheless, we found slight improvements in rotation, as well as in translation accuracy by simply including adversarial training into a camera pose regression framework due to better and more stable convergence of the model.

8.5.2 Pose Refinement

As a second step, we evaluate our proposed pose refinement strategy utilizing the trained discriminator network.

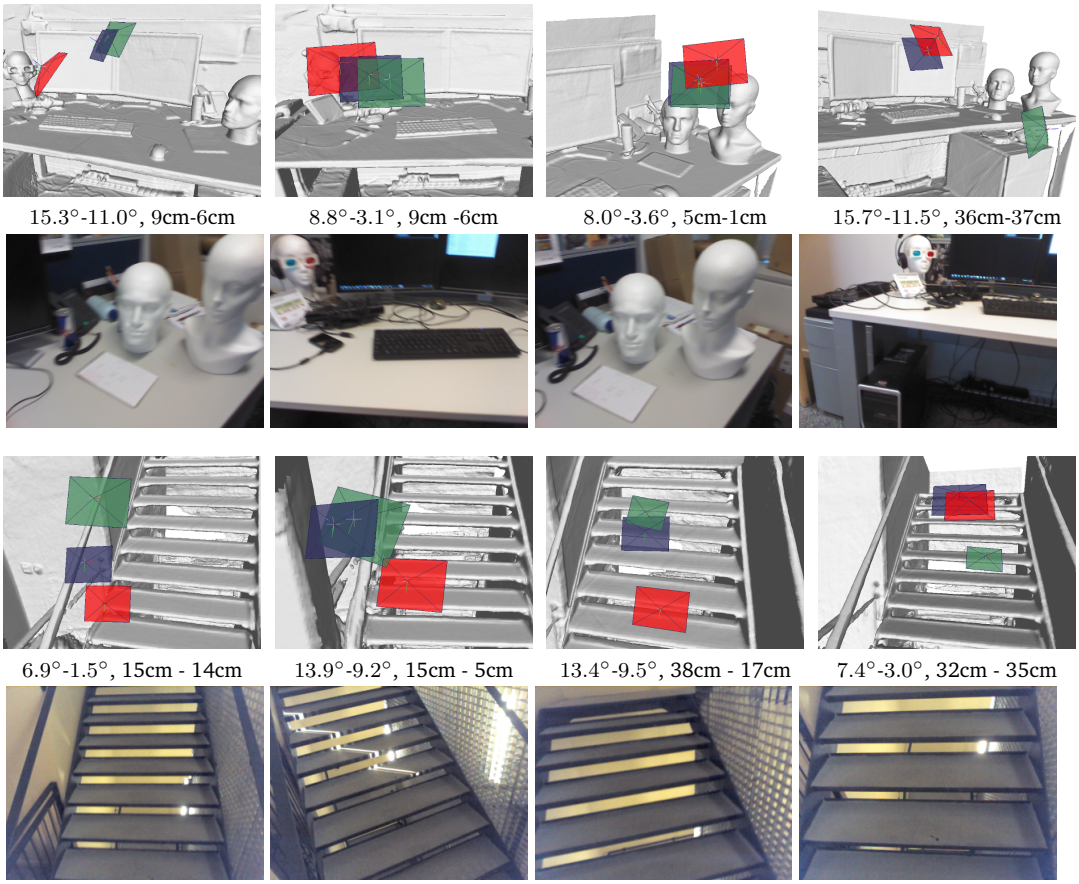
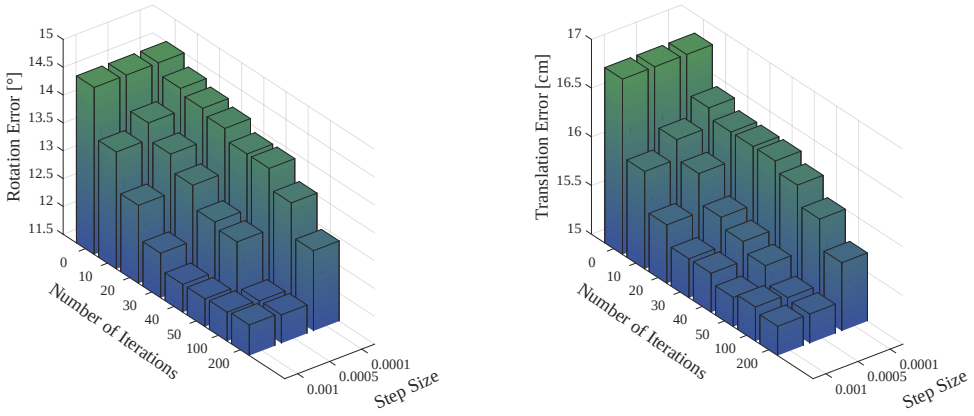


Fig. 8.4. RGB input images (second row) and the corresponding camera poses (first row), visualized in a reconstruction of the given scene. For each frame, the ground truth (green), initially regressed pose (red) and optimized pose using the proposed refinement (blue) are displayed. Below each visualization the respective rotation and translation errors before and after refinement are given.

Performance Evaluation

Surprisingly, even though the gradients coming from the discriminator have not specifically been trained to have geometric meaningful information, we can assume that to some extent this information has implicitly been encoded in the network. Therefore, we can use the gradients to update the regressed poses for any test image, given the constraints described in Section 8.4 on the quaternion update. Table 8.1 and Figure 8.3 summarize our findings. We report the median rotation and translation error in Table 8.1. Figure 8.3 shows the overall distribution of the aforementioned errors on the *Heads* scene of the 7-Scenes dataset. Overall we can report an improvement in pose accuracy by applying the proposed pose refinement. Further, we qualitatively assess the accuracy of our predictions, examples of which can be found in Figure 8.4. It can be seen both quantitatively and qualitatively that the regressed pose can effectively be pushed further towards the ground truth pose by the deep learning based refinement step. As as result, for example in comparison to our baseline method *Ours (log q)* we obtained a relative improvement in rotation of 12.0% and 31.1% for the *Heads* and *Stairs* scene respectively when using the proposed refinement (*Ours+Ref. (log q)*), see Table 8.4.



(a) Rotation Error **(b)** Translation Error

Fig. 8.5. Effect of different numbers of iterations as well as step sizes on the median rotation and translation errors for the proposed refinement, shown on the *Heads* scene. Our refinement can significantly improve the localization accuracy even in a few iterations of optimization.

Analysis of Optimization Parameters

Further, we investigate the effect of the step size l as well as the number of iterations on the localization accuracy of the proposed pose refinement. The results of our investigation are summarized in Figure 8.5, where we show median rotation and translational error on the *Heads* scenes for different numbers of refinement iterations as well as step sizes. A lower step size usually leads to smaller changes in the pose, but, therefore, can also require a higher number of iterations to converge to the desired pose. Since this optimization process is required during testing, increasing the number of iterations is directly proportional to an increase in computational time. Experiments with larger step sizes ($l > 10^{-3}$) resulted in deterioration of the camera poses due to the optimization procedure becoming unstable. Usually only a few iterations of refinement are sufficient, though, to improve the regressed poses and provide a good improvement in camera pose accuracy, whereas the run-time of RANSAC-based methods, for example, depends on the quality of correspondences found. As a trade-off, we chose the parameter setting described in Section 8.4. For example, on average the refinement has a computational time of 42ms for 30 iterations, but grows linearly with the number of iterations. Although we were able to achieve promising results with the proposed pose refinement strategy, it should be noted that it remains an optimization procedure itself, and thus depends on factors such as the quality of initialization. Therefore, in some cases the refinement might result in a solution that is not preferable to the initially regressed pose or difficult to recover from, if the predicted pose is far away from the ground truth one, an example of which is shown in Figure 8.4 d).

8.5.3 Influence of Feature Extractor

We have seen in Chapter 4 that a learned feature representation can have a high impact on the performance of deep learning models, especially for pose regression. Naturally, we assume the extracted feature descriptor of a query vector to possibly have a high influence on our model.

Tab. 8.2. Relative decrease, in percentage, of the median rotation and translation error after refinement in comparison to initially regressed poses. Evaluated are different network architectures used to obtain a feature representation of the RGB image input, showing the influence of the feature extractor on the proposed refinement. Higher values correspond to improved pose accuracy.

Heads	**Without** $f(\mathbf{X})$	**AlexNet** [112]	**VGG-16** [203]	**ResNet-18** [87]
Rotation	4.25%	3.56%	8.32%	12.18%
Translation	-3.0%	2.88%	4.7%	4.39%

Therefore, to evaluate the effect of the feature extraction network on the discriminator and subsequently the camera pose refinement, we evaluated our method using several different network architectures. We compare between AlexNet [112], VGG16 [203] and ResNet-18 [87] pre-trained on ImageNet for the task of image classification. For all models the refinement is run for thirty iterations.

Additionally we experiment with feeding only the regressed camera poses to train the discriminator network. For this experiment, to keep the complexity of the models on the same level, we replace the convolutional layers of the discriminator network with fully connected layers of roughly equal number of trainable parameters as the convolutional counterpart of the discriminator. Since by design of our method for each architecture we have to train a new model, we report the relative decrease in rotation and translation error over the initially regressed pose quality of the respective model. The results are summarized in Table 8.2. We found that our proposed refinement is fairly robust to the extracted features and were able to obtain improved pose accuracy regardless of the network architecture used, except when using pose information only, without additional information about the corresponding image representation. Nevertheless, we found an increase in localization performance depending on the choice of network architecture with the best performing model resulting in the ResNet-18 [87] network architecture.

8.5.4 Runtime Evaluation

In comparison to computationally expensive refinement strategies based on RANSAC, we are aiming to provide a reasonably fast RGB-based alternative. Therefore, we now report the computational requirements of our method, as well as of each individual part of our framework. Table 8.3 shows the computational times of the individual steps of our method evaluated for a single frame. Pose refinement is calculated for thirty iterations. The method is implemented in Python and PyTorch and run on a 11GB NVIDIA GeForce RTX 2080 graphics card and 64 GB Intel Core i7. To mention, state-of-the-art method such as [199] and [21] report run-times of one or two hundred milliseconds, although programmed in C++ and run on different systems.

Tab. 8.3. Computational times for our pipeline as well as each individual step, initial pose regression, feature extraction and subsequent iterative pose refinement for thirty iterations.

Pose regression	Feature extraction	Pose refinement	Overall
4.5ms	3ms	42ms	$\sim$ 50ms

Tab. 8.4. Comparison between recent state-of-the-art direct camera pose regression methods and our results without (Ours) and with pose refinement (Ours+Ref.). Following the state of the art, displayed is the median rotation and translation error in meter and degrees evaluated on the 7-Scenes dataset.

	Scene	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs	Average
	DSAC++ RGB [24]	0.7°, 0.02m	1.1°, 0.03m	6.7°, 0.12m	0.8°, 0.03m	1.1°, 0.05m	1.3°, 0.05m	5.1°, 0.29m	2.4°, 0.08m
	PoseNet RGB [105]	4.5°, 0.14m	11.8°, 0.27m	12.1°, 0.18m	5.7°, 0.20m	4.8°, 0.25m	5.5°, 0.24m	10.6°, 0.37m	7.9°, 0.24m
	MapNet [26]	4.2°, 0.11m	11.7°, 0.29m	13.1°, 0.20m	6.4°, 0.19m	5.8°, 0.23m	5.8°, 0.27m	12.4°, 0.31m	8.5°, 0.23m
	Ours	4.9°, 0.13m	11.0°, 0.30m	14.5°, 0.17m	6.7°, 0.22m	6.7°, 0.23m	5.9°, 0.27m	13.5°, 0.32m	9.0°, 0.23m
	Ours+Ref.	4.8°, 0.12m	10.2°, 0.29m	12.0°, 0.15m	6.6°, 0.21m	6.5°, 0.22m	5.8°, 0.26m	12.2°, 0.30m	8.3°, 0.22m
log q	MapNet [26]	4.3°, 0.11m	12.1°, 0.27m	12.2°, 0.19m	6.4°, 0.19m	5.1°, 0.22m	5.3°, 0.25m	11.3°, 0.30m	8.1°, 0.22m
log q	Ours	5.0°, 0.13m	11.8°, 0.28m	14.1°, 0.17cm	7.1°, 0.20m	5.4°, 0.22m	6.2°, 0.26m	12.2°, 0.29m	8.8°, 0.22m
log q	Ours+Ref.	4.8°, 0.12m	11.6°, 0.27m	12.4°, 0.16m	6.8°, 0.19m	5.2°, 0.21m	6.0°, 0.25m	8.4°, 0.28m	7.9°, 0.21m

8.5.5 Comparison to the State of the Art

As our main focus in this work is to investigate the effect of our proposed framework on direct camera pose regression methods that rely on RGB information only, we show a comparison to recent methods in this regard, namely PoseNet [105] and MapNet [26], which also forms our baseline model. We choose PoseNet and MapNet versions solely relying on single RGB input image. In comparison to the original PoseNet [107], [105] uses a more sophisticated loss function, which incorporates homoscedastic uncertainty and if available refines the trained network using a geometric re-projection loss function to further improve the regressed camera poses. However, due to our restriction to RGB methods we report the results of the method withour said geometric loss function. The results can be seen in Table 8.4. We evaluate both models trained to predict quaternions as well as the logarithm of quaternions to show the effectiveness of our method regardless of the baseline representation used. In comparison to both [105] and [26], we found overall improvements in pose accuracy using the proposed refinement, where the effect of our method seems to be most profound on scenes for which only a small number of training images is available, such as *Heads* and *Stairs*. In addition we include a recent scene coordinate regression method, DSAC++ [24], that given a constant depth prior, can be trained solely relying on RGB information as well. As can be seen, the regressed 3D information, and following pose refinement, greatly improve the accuracy of the predicted camera poses, which leads to the method outperforming direct camera pose regression methods and ours. This, however, comes at a significant drop in computational time. Lastly, although we focus on RGB only solutions in this paper, it should be mentioned that our core regression method could be easily extended to include further information, like relative pose information or geometric constraints as in [26].

8.6 Disscusion and Conclusion

In conclusion, we have presented a novel approach for camera re-localization applications solely relying on RGB information. Building on top of direct camera pose regression methods, we use the regressed camera poses and features extracted from the input image to train a discriminator network that tries to distinguish between generated and ground truth poses, and thus implicitly tries to learn the geometric connection between RGB image and the corresponding camera pose. We have analyzed each component of our framework to evaluate this assumption and were able to achieve promising results. Further, we proposed a novel RGB-based pose refinement, where we use the trained discriminator network to update and optimize the initially regressed poses, showing that the network can actually learn a meaningful representation of the camera poses and image space, and in turn can use this information to further improve localization accuracy.

Although our method obtained promising results, its accuracy in comparison to structure-based methods still remains significant. However, we believe that this gap is mostly due to the general problems direct regression methods still suffer from. First of all acquiring large datasets to train deep learning models for the task of camera localization still remains a challenging task. Most datasets currently rely on structure from motion or SLAM frameworks and use the computed camera poses as ground truth. However, this ground truth has its own associated error depending on the method and environment, a topic recently addressed in [242] as well. As a result most datasets are first of all not sufficiently accurate in terms of ground truth and secondly contain few training images, at most a couple of thousands, in comparison to large scale datasets like ImageNet [184], with millions of images, obtained for classification tasks.

Naturally as a result direct regression methods tend to overfit to the training set or show little generalization capabilities when given a query image far from the training set's camera trajectories [193]. In turn, we believe that addressing these issues will significantly improve our method's performance and leave this as a future task.

A second aspect, that we would like to analyze within the scope of this thesis, is how direct regression methods perform within highly ambiguous environments. As addressed in Chapter 6, outliers or erroneous predictions can highly impact the performance of a model. Erroneous predictions can occur through the uncertainty of a model itself as well as due to noisy or ambiguous input, arising from repetitive structures and symmetries in the scene, and can be a result of wrong matches or image similarities between viewpoints.

9 Uncertainty-Aware Multimodal Pose Regression

9.1 Motivation

Direct camera pose regression methods have shown low cost and computationally fast solutions to solving the camera pose estimation problem solely relying on RGB information. However, due to the lack of 3D information, these methods still suffer in generalization capabilities. In addition, similar to their feature-based counterparts, repetitive structures and symmetries arising in the environments, can easily degrade their performance. Visually similar images, even though captured from highly different viewpoints, as depicted in Figure 9.1, are problematic to effectively be localized.

In these cases, a single solution, as most current methods predict, might not exist. If not able to reliably predict a pose, a measure of uncertainty in its prediction provides necessary information to prevent system failure or at least provide feedback to the system's user. In the next sections, we therefore address the task of uncertainty estimation in direct camera pose regression methods.

Additionally, ambiguities arising in the scene, have to be handled accordingly. If not able to predict a single correct solution, the range of correct solutions can be provided to identify and resolve ambiguous queries.

For this aim, we propose a framework that is able to predict an uncertainty estimate for a pose prediction by predicting a continuous distribution over the pose space such that the distributions variance is correlated to the pose's uncertainty. Further, we propose to predict a

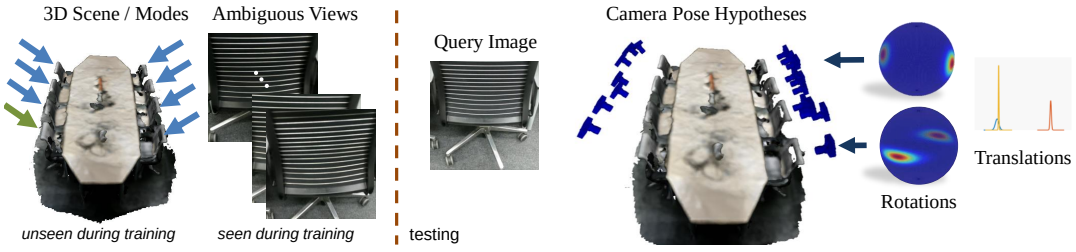


Fig. 9.1. In a highly ambiguous environment, similar looking views can easily confuse current camera pose regression models and lead to incorrect localization results. Instead, given a query RGB image, our aim is to predict the possible modes as well as the associated uncertainties, which we model by the parameters of Bingham and Gaussian mixture models.

range of solutions by incorporating a multiple hypotheses training scheme into our framework that is able to capture ambiguous views and their corresponding camera poses.

A significant amount of the work presented here is part of the following publication

'6D Camera Relocalization in Ambiguous Scenes via Continuous Multimodal Inference' by Mai Bui, Tolga Birdal, Haowen Deng, Shadi Albarqouni, Leonidas Guibas, Slobodan Ilic and Nassir Navab, 2020 [34].

9.2 Related Work

First, we briefly describe the related work with regard to our method. Object pose estimation is a highly related field with similar issues arising in this context as well. In this application, object symmetries have shown to be a main source of ambiguity. Therefore we now highlight recent research in this area, before moving to uncertainty estimation methods as well as multiple hypotheses solutions.

Handling Object Symmetries There has been a significant amount of research addressing object symmetries with respect to object pose estimation and how to handle ambiguous views arising due to these symmetries [51, 104, 169, 173].
Most methods aim at restricting the pose space for symmetric objects, such that ambiguous labels are avoided during training [104, 236]. BB8 by Rad and Lepetit [173] for example restricts the pose space seen during training according to the angle of symmetry of the corresponding object and further includes a classification step to handle borderline cases.
On the other hand, Corona et al. [51] specifically include multiple correct labels for symmetric objects in their loss function to handle ambiguities and further classify the degree of symmetry for a given object, which is inferred from rendered depth images of the object. These methods mainly rely on prior information in the form of a CAD model, which especially for many industrial applications is a valid approach. In the context of camera localization, such prior knowledge of the scene is normally not available and would be highly problematic to acquire. Therefore, it is not straightforward to apply or adapt such methods to the task addressed in this thesis.

Multiple Hypotheses Estimation On the other hand, instead of inferring a single pose estimate, allowing a range of possible solutions has recently become a promising research direction. Rupprecht et al. [182] show a general framework that extends a simple regression model to output multiple hypotheses that correspond to a Voronoi tessellation. The method is easily applicable to a range of applications such as human pose estimation or instance segmentation.

Further, Manhardt et al. [135] show that multiple hypotheses prediction can be used as an indication of possible ambiguous cases in object pose estimation. By obtaining multiple object orientation hypotheses and fitting a Bingham distribution to these samples, [135] are using the predicted distribution and its variance to identify ambiguous views and symmetries of the object.

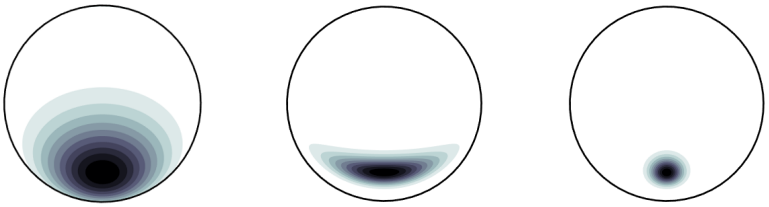
On top of multiple hypothesis prediction, Makansi et al. [133] fit a mixture model to the resulting hypothesis, obtaining a predicted, but not directly learned, mixture distribution and apply their method to movement prediction in autonomous driving scenarios. Neither of the above methods, however, apply their model to the task at hand, camera localization.

Uncertainty Estimation Typical convolutional neural networks [87, 202] are over-confident in their predictions [83, 249] and tend to approximate the conditional averages of the target data [15]. These undesired properties render the immediate outputs of those networks unsuitable for the quantification of uncertainty. This has fostered numerous works as we will summarize in the following. *Mixture Density Networks (MDN)* [15], introduced by Christopher Bishop, is the pioneer to model the conditional distribution by predicting the parameters of a Gaussian mixture model. It has been successfully applied in a wide range of applications such as object detection and localization [153], classification of image parts [120] as well as speech synthesis [240].

Yet, it is repeatedly reported that optimizing for general mixture models suffers from mode collapse and numerical instabilities with increasing dimensionality [53, 133]. These issues can to a certain extent be addressed by using Dropout [68] as a Bayesian approximation, but even for moderate dimensions these methods still face difficulties in capturing multiple modes and, due to the sampling procedure applied, have a higher computational complexity.

The problem of the model's uncertainty in the context of predicting camera poses has less often been addressed [38, 50, 106]. Initial attempts to capture the uncertainty of camera re-localization involved random forests [28]. Valentin et al. [220] stored components of a Gaussian mixture model at the leaves of a scene coordinate regression forest [199]. The modes are obtained via a mean shift procedure, and the covariance is explained by a 3D Gaussian. A similar approach later considered the uncertainty in object coordinate labels [22]. It is a shortcoming of random forests that both of these approaches require hand crafted depth features. Moreover, their uncertainty is on the correspondences and not on the final camera pose. Thus a costly RANSAC [66] is required to propagate the uncertainty in the leaves to the camera pose. In [106] the authors of the original PoseNet model extend their approach as a Bayesian convolutional neural network and use dropout to compute an uncertainty measure as the variance of the network's prediction. This requires repeatedly evaluating the network for one image to obtain a distribution over possible camera poses. As a result, the method only provides an uncertainty measure for the final pose prediction, whereas we aim at providing an uncertainty estimate for each sample pose as well. Similar, Huang et al. [94] use Dropout to remove the influence of features or pixels belonging to dynamic objects, such as pedestrians, that are not reliable to infer the cameras pose. Although the method shows promising improvements, it does not capture the uncertainty of a camera pose estimate either.

In contrast VidLoc [50] re-formulates the problem as Gaussian mixture regression and uses a mixture density network [15] to additionally predict the model's uncertainty as the variance of the resulting mixture model. The authors of [50] do not address the well-known problem of mode collapse, that mixture density networks suffer from. Further, a Gaussian distribution is not well suited to model the orientation of a camera when represented as a quaternion. Instead we propose to model the orientation of a camera using the Bingham distribution,



(a) $\lambda_1 = -3, \lambda_2 = -3$ **(b)** $\lambda_1 = -4, \lambda_2 = -30$ **(c)** $\lambda_1 = -40, \lambda_2 = -30$

Fig. 9.2. Example 2D visualizations of Bingham distributions for varying concentration parameters λ.

whose qualities perfectly fit to model the properties of quaternions. Therefore, we first start with describing the properties and qualities of the Bingham distribution in detail.

9.3 The Bingham Distribution

The Bingham distribution [13] is derived from a zero-mean Gaussian conditioned to lie on the unit sphere $\mathbb{S}^{d-1}$. Its probability density function is defined as $\mathcal{B} : \mathbb{S}^{d-1} \to R$:

$$\mathcal{B}(\mathbf{x}; \mathbf{\Lambda}, \mathbf{V}) = \frac{1}{F} \exp(\mathbf{x}^T \mathbf{V} \mathbf{\Lambda} \mathbf{V}^T \mathbf{x}) \tag{9.1}$$

$$= \frac{1}{F} \exp\left(\sum_{i=1}^{d} \lambda_i (\mathbf{v}_i^T \mathbf{x})^2\right). \tag{9.2}$$

In this equation $\mathbf{V} \in \mathbb{R}^{d \times d}$ is an orthogonal matrix ($\mathbf{V}\mathbf{V}^T = \mathbf{V}^T\mathbf{V} = \mathbf{I}_{d \times d}$) describing the orientation and $\mathbf{\Lambda} \in \mathbb{R}^{d \times d}$ is called the *concentration matrix* with $0 \geq \lambda_1 \geq \cdots \geq \lambda_{d-1}$ with

$$\mathbf{\Lambda} = \begin{bmatrix} 0, & & & & \\ & \lambda_1 & & & \\ & & \lambda_2 & & \\ & & & \ddots & \\ & & & & \lambda_{d-1} \end{bmatrix}_{d \times d} . \tag{9.3}$$

For illustration of the distribution example 2D visualizations of a Bingham distribution in $\mathbb{S}^2$ for varying concentration matrices can be seen in Figure 9.2, which visualized a mapping of the probability density to the unit sphere. The concentration matrix indicates the variance of the Bingham distribution, where high negative values correspond to less variance of the distribution. In our work, we use the concentration values to derive a measure of uncertainty in a neural network's prediction.

It can be shown that adding a multiple of the identity matrix $\mathbf{I}_{d \times d}$ to $\mathbf{V}$ does not change the distribution [13]. Moreover, it is possible to swap the columns of $\mathbf{\Lambda}$ without changing the distribution as long as the same permutation is applied to $\mathbf{V}$. therefore, we can build $\mathbf{V}$ in a

sorted fashion and conveniently force the first entry of $\mathbf{\Lambda}$ to be zero. This allows us to obtain *the mode* very easily by taking the first column of $\mathbf{V}$.

F in 9.1 denotes the *the normalization constant* dependent only on $\mathbf{\Lambda}$ and is of the form:

$$F \triangleq |S_{d-1}| \cdot {}_1F_1\Big(\frac{1}{2}, \frac{d}{2}, \Lambda\Big), \tag{9.4}$$

where $|S_{d-1}|$ is the surface area of the d-sphere and ${}_1F_1$ is a confluent hypergeometric function of matrix argument [88, 115]. The computation of the normalization constant is complex and computationally expensive, however it can be simplified to be independent of $\mathbf{V}$. Being only a function of $\mathbf{\Lambda}$, in practice, we can approximate F by using a pre-computed look-up table and interpolation. Likewise, to enable backpropagation through a neural network, the look-up table can in turn be used to approximate the gradients of F with respect to $\mathbf{\Lambda}$ [113, 116].

Relationship to quaternions. The Bingham distribution is an antipodally symmetric probability distribution. Due to this property it is well suited to explain the topology of quaternions, i. e., $\mathcal{B}(\mathbf{x}; \cdot) = \mathcal{B}(-\mathbf{x}; \cdot)$ holds for all $\mathbf{x} \in \mathbb{S}^{d-1}$. Bingham distributions have been extensively used to represent distributions on quaternions [76, 77, 115]; however, to the best of our knowledge, never for the problem we consider here.

Constructing a Bingham distribution on a given mode Given a quaternion $\mathbf{q}$, there are multiple ways to construct a corresponding Bingham distribution, in particular in constructing $\mathbf{V}$. In this thesis, we evaluate three methods, showing different properties in terms of parameters and their use with neural networks. First, we follow Birdal et al. [14]. Since creating a Bingham distribution on any given mode $\mathbf{q} \in \mathbb{R}^4$ requires finding a set of vectors orthonormal to $\mathbf{q}$, we use the *parallelizability* ($d = 1, 2, 4$ or 8) of unit quaternions to define the orthonormal basis $\mathbf{V} : \mathbb{R}^4 \mapsto \mathbb{R}^{4\times4}$ as follows:

$$\mathbf{V}(\mathbf{q}) \triangleq \begin{bmatrix} q_1 & -q_2 & -q_3 & q_4 \\ q_2 & q_1 & q_4 & q_3 \\ q_3 & -q_4 & q_1 & -q_2 \\ q_4 & q_3 & -q_2 & -q_1 \end{bmatrix}. \tag{9.5}$$

This results in a matrix composed of four unit vectors: the mode and its orthonormals. The authors show that it is easy to verify that the matrix valued function $\mathbf{V}(\mathbf{q})$ is orthonormal for every $\mathbf{q} \in \mathbb{R}^4$. $\mathbf{V}(\mathbf{q})$ further gives a convenient way to represent quaternions as matrices paving the way to linear operations, such as quaternion multiplication.

Alternatively, Gram-Schmidt can be used to compute an orthonormal matrix $\mathbf{V}$ for a given matrix $\mathbf{M} \in \mathbb{R}^{d\times d}$, where the column vectors $\mathbf{v_i}$ of $\mathbf{V}$ are computed from the column vectors $\mathbf{m_i}$ as follows

$$\hat{\mathbf{v}}_\mathbf{i} = \mathbf{m}_i - \sum_{k=1}^{i-1} \langle \mathbf{v}_k, \mathbf{m}_i \rangle \cdot \mathbf{v}_k \text{ , where } \mathbf{v}_i = \frac{\hat{\mathbf{v}}_i}{\|\hat{\mathbf{v}}_i\|}. \tag{9.6}$$

In addition we propose the following procedure using the *Cayley transform*, which describes the mapping from skew-symmetric matrices to special orthogonal matrices. Given a vector $\mathbf{q}$ (in this case not necessarily with unit norm), we compute $\mathbf{V}$ as

$$\mathbf{V} = (\mathbf{I}_{4\times4} - \mathbf{S})^{-1}(\mathbf{I}_{4\times4} + \mathbf{S}), \tag{9.7}$$

where $\mathbf{I}_{4\times4}$ is the identity matrix and

$$\mathbf{S}(\mathbf{q}) \triangleq \begin{bmatrix} 0 & -q_1 & q_4 & q_3 \\ q_1 & 0 & q_3 & q_2 \\ -q_4 & -q_3 & 0 & -q_1 \\ q_3 & q_2 & q_1 & 0 \end{bmatrix}. \tag{9.8}$$

a skew-symmetric matrix of $\mathbf{q}$. As $\mathbf{S}$ is a skew-symmetric matrix, $\mathbf{I}_{4\times4} - \mathbf{S}$ is invertible and the proposed mapping results in a orthogonal matrix $\mathbf{V}$ with $\det(\mathbf{V}) = 1$.

Relationship to other representations Note that geometric [7] or measure theoretic [65], there are multitudes of ways of defining probability distributions on the Lie group of 6D rigid transformations. A naive choice would be to define Gaussian distribution on the Rodrigues vector (or exponential coordinates) [154] where the geodesics are straight lines [150]. However, as our purpose is direct regression, in this work we favor quaternions as continuous and minimally redundant parameterizations without singularities [80] and use the Bingham distribution that is well suited to their topology. We handle the redundancy $\mathbf{q} \equiv -\mathbf{q}$ by mapping all the rotations to the northern hemisphere.

9.4 Continuous Multimodal Inference

We now describe our model for uncertainty prediction following [73, 170]. We consider the situation where we observe an input image $\mathbf{X} \in \mathbb{R}^{W\times H\times3}$, of width W and height H, and assume the availability of a predictor function $\mu_{\mathbf{\Gamma}}(\mathbf{X}) : \mathbb{R}^{W\times H\times3} \mapsto \mathbb{R}^4$ parameterized by $\mathbf{\Gamma}$.

The unimodal case We momentarily assume that $\mu_{\mathbf{\Gamma}}(\cdot)$ can yield the correct values of the absolute camera rotation $\mathbf{q} \in \mathbb{R}^4$ with respect to a common origin, admitting a non-ambiguous prediction, hence a posterior of single mode. We use the predicted rotation to set the most likely value, the mode, of a Bingham distribution:

$$p_{\mathbf{\Gamma}}(\mathbf{q} \,|\, \mathbf{X}; \mathbf{\Lambda}) = \frac{1}{F}\exp\left(\mathbf{q}^\top \mathbf{V}_\mu \mathbf{\Lambda} \mathbf{V}_\mu^\top \mathbf{q}\right), \tag{9.9}$$

and let $\mathbf{q}$ differ from this value up to the extent determined by $\mathbf{\Lambda} = \{\lambda_i\}$. For the sake of brevity we use $\mathbf{V}_\mu \equiv \mathbf{V}(\mu_{\mathbf{\Gamma}}(\mathbf{X}))$, the orthonormal basis aligned with the predicted quaternion $\mu_{\mathbf{\Gamma}}(\mathbf{X})$ and as defined in 9.5.

While for certain applications, fixing $\mathbf{\Lambda}$ can work, in order to capture the variation in the input, it is recommended to adapt $\mathbf{\Lambda}$ [170]. Thus, we introduce it among the unknowns. To this end we define the function $\Lambda_{\mathbf{\Gamma}}(\mathbf{X})$ or in short $\mathbf{\Lambda}_{\mathbf{\Gamma}}$ for computing the concentration values

depending on the current image and the parameters $\mathbf{\Gamma}$. Our final model for the unimodal case reads:

$$p_{\mathbf{\Gamma}}(\mathbf{q} \,|\, \mathbf{X}) = \frac{\exp\left(\mathbf{q}^\top \mathbf{V}(\mu(\mathbf{X}))\Lambda_{\mathbf{\Gamma}}(\mathbf{X})\mathbf{V}(\mu(\mathbf{X}))^\top \mathbf{q}\right)}{F(\Lambda_{\mathbf{\Gamma}}(\mathbf{X}))} \tag{9.10}$$

$$= \frac{\exp\left(\mathbf{q}^\top \mathbf{V}_\mu \Lambda_{\mathbf{\Gamma}} \mathbf{V}_\mu^\top \mathbf{q}\right)}{F(\Lambda_{\mathbf{\Gamma}})}. \tag{9.11}$$

The second row follows from the short-hand notations and is included for clarity. Given a collection of observations i.e., images $\mathcal{X} = \{\mathbf{X}_i\}$ and associated rotations $\mathbf{Q} = \{\mathbf{q}_i\}$, the parameters of $\mu_{\mathbf{\Gamma}}(\mathbf{X})$ and $\Lambda_\Gamma(\mathbf{X})$ can be obtained simply by maximizing the log-likelihood:

$$\mathbf{\Gamma}^\star = \arg\max_{\mathbf{\Gamma}} \; \log \mathcal{L}_u(\mathbf{\Gamma}|\mathcal{X}, \mathbf{Q}) \tag{9.12}$$

$$\log \mathcal{L}_u(\mathbf{\Gamma}|\mathcal{X}, \mathbf{Q}) = \sum_{i=1}^{N} \mathbf{q}_i^\top \mathbf{V}_{\boldsymbol{\mu}} \Lambda_{\mathbf{\Gamma}} \mathbf{V}_{\boldsymbol{\mu}}^\top \mathbf{q}_i - \sum_{i=1}^{N} \log F\big(\Lambda_{\mathbf{\Gamma}}\big).$$

If $\Lambda_{\mathbf{\Gamma}}$ were to be fixed as in [170], the term on the right would have no effect and minimizing that loss would correspond to optimizing the Bingham log-likelihood. To ensure $0 \geq \lambda_1 \geq \cdots \geq \lambda_{d-1}$, we predict λ_1 and offsets $e_2, ..., e_{d-1}$ for the remaining concentration parameters, such that

$$\lambda_2 = \lambda_1 - e_2, ..., \lambda_{d-1} = \lambda_{d-2} - e_{d-1} \tag{9.13}$$

holds true.

Extension to finite Bingham Mixture Models (BMM) Ambiguities present in the data requires us to take into account the multimodal nature of the posterior. To achieve this, we now extend the aforementioned model to Bingham Mixture Models [177]. For the finite case, we use K different components associated with K mixture weights $\pi_j(\mathbf{X}, \mathbf{\Gamma})$ for $j = 1, ..., K$. With each component being a Bingham distribution, we can describe the density function as

$$p_{\mathbf{\Gamma}}(\mathbf{q}_i \,|\, \mathbf{X}_i) = \sum_{j=1}^{K} \pi_j(\mathbf{X}_i, \mathbf{\Gamma}) p_{\mathbf{\Gamma}j}(\mathbf{q}_i \,|\, \mathbf{X}_i), \tag{9.14}$$

where $p_{\mathbf{\Gamma}j}(\mathbf{q}_i \,|\, \mathbf{X}_i)$ are the K component distributions and $\pi_j(\mathbf{X}_i, \mathbf{\Gamma})$ the mixture weights s.t. $\sum_j \pi_j(\mathbf{X}_i, \mathbf{\Gamma}) = 1$. The model can again be trained by maximizing the log-likelihood, but this time of the mixture model [123, 230]:

$$\mathbf{\Gamma}^\star = \arg\max_{\mathbf{\Gamma}} \; \log \mathcal{L}_m(\mathbf{\Gamma}|\mathcal{X}, \mathbf{Q}) \tag{9.15}$$

$$\log \mathcal{L}_m(\mathbf{\Gamma}|\mathcal{X}, \mathbf{Q}) = \sum_{i=1}^{N} \log \sum_{j=1}^{K} \pi_j(\mathbf{X}_i, \mathbf{\Gamma}) p_{\mathbf{\Gamma}j}(\mathbf{q}_i \,|\, \mathbf{X}_i).$$

Deeply modeling $\mu(\cdot)$ and $\Lambda(\cdot)$ Following up on the recent advances, we jointly model $\mu(\cdot)$ and $\Lambda(\cdot)$ by a deep residual network [87]. $\mathbf{\Gamma}$ denotes the entirety of the trainable parameters. On the output we have eight quantities per Bingham density: four for the mode quaternion, three for $\mathbf{\Lambda}$ and one for the weight $\pi_j(\cdot)$. In total, as we have K mixture components, resulting in $K \times 8$ output entities. While a typical way to train our network is through simultaneously regressing the output variables, this is known to severely harm the accuracy [182]. Instead

we exploit modern approaches to training in presence of ambiguities as we detail in what follows.

MHP training scheme Due to the increased dimensionality, in practice training our variational network in an unconstrained manner is likely to suffer from mode collapse, where all the heads concentrate around the same prediction. To avoid this and obtain a diverse set of modes, instead of training all branches equally by maximizing the log-likelihood of the mixture model, we follow the multi hypotheses schemes of [133, 182] and train our model using a Winner-Takes-All loss function, for each branch maximizing the log-likelihood of a unimodal distribution,

$$\mathbf{\Gamma}^{\star} = \arg\max_{\mathbf{\Gamma}} \sum_{i=1}^{N} \sum_{j=1}^{K} w_{ij} \log \mathcal{L}_u(\mathbf{\Gamma}|\mathbf{X}_i, \mathbf{q}_i)), \tag{9.16}$$

according to the associated weights w_{ij} for each of the k hypotheses. In this work, we compute the weights w_{ij} during training following RWTA [182] where

$$w_{ij} = \begin{cases} 1-\epsilon, & \text{if } j = \arg\min_k d(\mathbf{q}_i, \hat{\mathbf{q}}_{ik}) \\ \frac{\epsilon}{K-1}, & \text{otherwise} \end{cases}, \tag{9.17}$$

for a given distance function $d(\cdot)$. Most often $d(\cdot)$ corresponds to the l_1 or l_2 norm of difference between the input vectors, given as

$$d(\mathbf{q}, \hat{\mathbf{q}}) = \|\mathbf{q} - \hat{\mathbf{q}}\|_*. \tag{9.18}$$

However, given that we are aiming to optimize the log likelihood of our model, it would most certainly be possible to choose the best branch on different measures, such as its loglikelihood, such that

$$d(\mathbf{q}, \hat{\mathbf{q}}) = -\log\, p_{\mathbf{\Gamma}}(\mathbf{q} \,|\, \mathbf{X}). \tag{9.19}$$

Note that WTA [85] would amount to updating only the branch of the best hypothesis and EWTA [133] the top k branches closest to the ground truth. However, for our problem, we found RWTA to be a more reliable machinery.

Finally, we compare between two models. First, we train the our BMM model as described in 9.14. However to stabilize training and avoid mode collapse, we add our MHP training scheme and corresponding loss. Note that, in this case, the mixture coefficient $\pi_j(\cdot)$ are trained implicitly by the BMM component. This results in the following loss functions used to train our neural network

$$\mathbf{\Gamma}^{\star} = \arg\max_{\mathbf{\Gamma}}(\ \log \mathcal{L}_m(\mathbf{\Gamma}|\mathcal{X}, \mathbf{Q}) + \boldsymbol{w} \log \mathcal{L}_u(\mathbf{\Gamma}|\mathcal{X}, \mathbf{Q})), \tag{9.20}$$

$$\tag{9.21}$$

such that the mixture coefficients are trained by our Bingham mixture model loss. In addition, for stability and diverse predictions, the branches are updated by our unimodal loss according to weights $\boldsymbol{w} = \{w_{ij}\}$, which define the best branches during training.

Second, to obtain the desired continuous distribution, we explicitly train the weights of our Bingham mixture model using the following loss function:

$$\mathcal{L}_\pi(\mathbf{\Gamma}|\mathcal{X},\mathbf{Q}) = \sum_{i=1}^{N}\sum_{j=1}^{K}\sigma(\hat{\pi}_j(\mathbf{X}_i,\mathbf{\Gamma}),y_{ij}), \tag{9.22}$$

where $\sigma(\cdot)$ is the cross-entropy, $\hat{\pi}(\cdot)$ the predicted weight of the neural network and y_{ij} the associated label of the mixture model component given as

$$y_{ij} = \begin{cases} 1, & \text{if } j = \arg\min_k d(\mathbf{q}_i,\hat{\mathbf{q}}_{ik}) \\ 0, & \text{otherwise} \end{cases}, \tag{9.23}$$

where $\hat{\mathbf{q}}_{ik}$ is the predicted mode of a single Bingham distribution. Our final loss, therefore, consists of the weighted likelihood for a unimodal distribution of each branch and the loss of our mixture weights, $\mathcal{L}_\pi(\mathbf{\Gamma}|\mathcal{X},\mathbf{Q})$:

$$\mathbf{\Gamma}^\star = \arg\min_{\mathbf{\Gamma}}(\mathcal{L}_\pi(\mathbf{\Gamma}|\mathcal{X},\mathbf{Q}) - \boldsymbol{w}\log\mathcal{L}_u(\mathbf{\Gamma}|\mathcal{X},\mathbf{Q}))). \tag{9.24}$$

$$\tag{9.25}$$

Inference Rather than reporting the conditional average which can result in label blur, we propose to obtain a single best estimate according to the weighted mode, where we choose the best mixture component according to its weight and pick the mode as a final prediction.

Definition of Uncertainty The entropy of a Bingham distribution is defined as

$$H_B = \log F - \mathbf{\Lambda}\frac{\nabla F(\mathbf{\Lambda})}{F}. \tag{9.26}$$

In the following we use the entropy of a Bingham distribution as a measure of uncertainty of a model in its prediction.

9.5 Evaluation Tools

For better analysis of our models, we specifically choose tools to inspect predicted Bingham distributions as well as uncertainty estimation. Throughout the next sections, we will evaluate our models using these tools and therefore briefly describe them here.

Sparsification Plot To analyze the performance of our models in terms of uncertainty estimation, we use what we call a *sparsification plot*. For this aim, we gradually remove the most uncertain samples, based for example on the predicted entropy and plot the mean rotation error of the remaining samples. This results in mean rotation errors plotted against the ratio of remaining samples used for evaluation. An example of such plot can be found in Figure 9.4.

Bingham Distribution Plots For each possible quaternion on the unit sphere, we evaluate the probability density of the sample using the predicted distribution parameters, $\mathbf{\Lambda}$ and $\mathbf{V}$, of our

(a) Object ID 01 (b) Object ID 02 (c) Object ID 03 (d) Object ID 15 (e) Object ID 30

Fig. 9.3. Subset of the objects from the TLess dataset used for evaluating our model on orientation estimation.

model. We marginalize the probability density over the angular component of the quaternion for each axis on the unit sphere. The values are then color coded, resulting in a distribution plot of the predicted Bingham distribution. Further, the resulting plot gives an intuition about the variance of the distribution. An example can be found in Figure 9.5.

9.6 Orientation Estimation

Before extending our model to incorporate translations and predict the full 6 DoF of a camera pose, we evaluate the proposed model on orientation estimation in ambiguous settings.

Dataset For this aim we use the T-Less [92] dataset, which was created for the purpose of evaluating object pose estimation methods. The dataset contains thirty industrial objects, and poses very challenging scenarios as the objects are texture-less, often symmetric and can be contained as parts of each other. However, these challenges make the dataset a perfect fit to evaluate our method, which is designed to handle ambiguities and object symmetries.

The dataset provides training images for each object, containing rendered views of the objects on black background sampled from a sphere with discrete viewpoints. The sampling results in around 1300 images for each object. Images are provided for three different sensors, a Primesense Carmine, Microsoft Kinect v2 and a Canon IXUS, out of which we chose the Microsoft Kinect version. We evaluate our method on five objects of the dataset, which can be seen in Figure 9.3, train object-specific models on a subset of the training images and use the remaining images (around 17%) for evaluation.

Evaluation on Uncertainty Estimation First, we evaluate our method on uncertainty estimation and train a neural network to predict a unimodal Bingham distribution for each sample. Figure 9.4 depicts the results of our analysis. Given highly ambiguous views from rotational symmetries our model predicts Bingham distributions accordingly. Depending on the level and axis of symmetry, this results in a high variance of the distribution. In case of non-ambiguous views that can be resolved by structural details of the object (last view in Figure 9.4), the orientation can be predicted with high certainty. Further, we analyze the correlation between rotation error and uncertainty with a sparsification plot, see Figure 9.5, showing high correlation between rotation error and uncertainty.

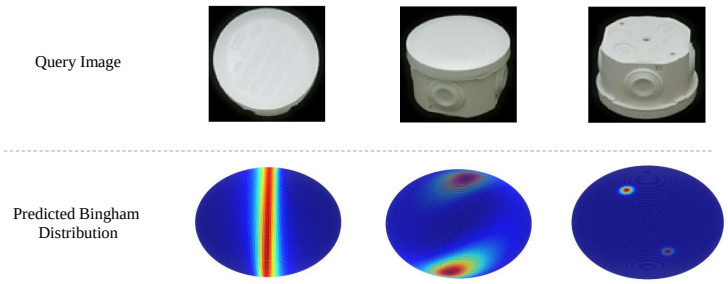
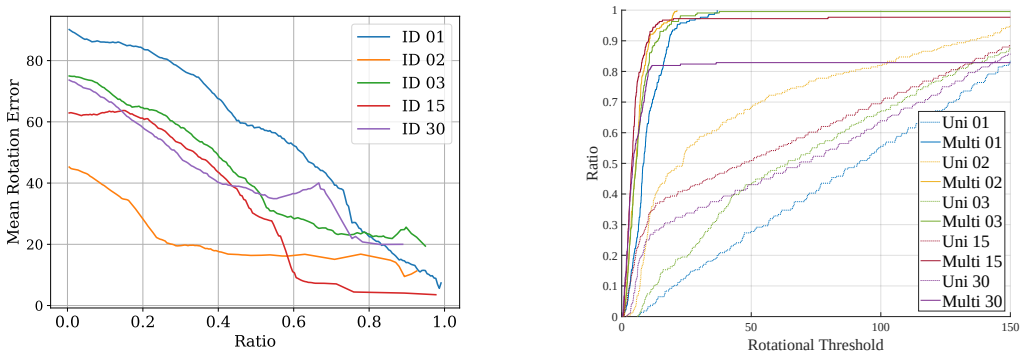


Fig. 9.4. Predicted Bingham distributions of our unimodal modal. The resulting predictions correlate with the level of uncertainty or symmetry of the object. Symmetries and ambiguous views result in high uncertainty in the corresponding rotation whereas structural details can lead to non-ambiguous views that the model can resolve and predict with high certainty.



(a) Sparsification Plot

(b) Single versus Mutimodal

Fig. 9.5. a) Uncertainty analysis on the five objects of the TLess dataset. By gradually removing samples with high entropy, a corresponding decrease in mean angular error of the predictions can be observed. b) Analysis of single versus multimodal model predictions, showing the ratio of correctly estimated rotations (y-axis) corresponding to various thresholds in degrees (x-axis).

Multimodal Inference A unimodal model can estimate uncertainties but is not able to handle symmetric objects and resulting ambiguous views properly. We therefore introduce a multimodal mixture model trained with multiple hypothesis prediction for stable training and without succumbing to mode collapse. Figure 9.5 shows the ratio of samples correctly predicted by our models for various thresholds used to define when a sample is correctly predicted. We compare between *unimodal* predictions and explicitly learning the mixture coefficient (see Equation 9.24).

9.7 Camera Pose Estimation

As a camera's pose is defined by its orientation as well as its position, we predict the rotation by our proposed Bingham Model and now describe in more detail how to model translations.

Incorporating translations Predicting entities that are non-Euclidean easily generalizes to the prediction of Euclidean quantities such as translations e.g. $\mathbf{t} \in \mathbb{R}^3$. Similar to the

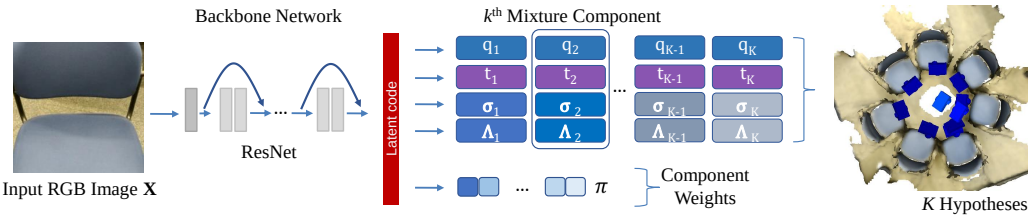


Fig. 9.6. Forward pass of our network. For an input RGB image we predict K camera pose hypotheses as well as Bingham concentration parameters, Gaussian variances and component weights to obtain a mixture model.

previously introduced model, we describe translations by a multivariate Gaussian distribution and model them using mixture density networks [15]. In more detail, for a sample input image $\mathbf{X} \in \mathbb{R}^{W \times H \times 3}$, we obtain a predicted translation $\hat{\mathbf{t}} \in \mathbb{R}^{c=3}$ from a neural network with parameters $\boldsymbol{\Gamma}$. This prediction is set to the most likely value of a multivariate Gaussian distribution with covariance matrix

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_c^2 \end{bmatrix}_{c \times c}, \tag{9.27}$$

where $\boldsymbol{\sigma}^2$ is predicted by our model. As a result our model for a unimodal Gaussian is defined as:

$$p_{\boldsymbol{\Gamma}}(\mathbf{t} \mid \mathbf{X}) = \frac{\exp(-\frac{1}{2}(\mathbf{t} - \hat{\mathbf{t}})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{t} - \hat{\mathbf{t}}))}{(2\pi)^{c/2}|\boldsymbol{\Sigma}|^{1/2}}, \tag{9.28}$$

where $c = 3$ and both $\hat{\mathbf{t}}$ as well as $\boldsymbol{\Sigma}$ are trained by maximizing its log-likelihood.

Similar to forming a Bingham Mixture Model, we can equally compute a Gaussian Mixture Model with K components and corresponding weights $\boldsymbol{\pi}(\mathbf{X}, \boldsymbol{\Gamma})$, such that $\sum_{j=1}^{K} \pi_j(\mathbf{X}, \boldsymbol{\Gamma}) = 1$, to obtain a multi-modal solution. Again both $\hat{\mathbf{t}}$ and $\boldsymbol{\Sigma}$ as well as $\boldsymbol{\pi}(\mathbf{X}, \boldsymbol{\Gamma})$ are learned by the network and trained by maximizing the log-likelihood of the mixture model. Note that, in this case, the components of $\hat{\mathbf{t}}$ are assumed to be statistically independent within each distribution component. However, it has been shown that any density function can be approximated up to a certain error by a multivariate Gaussian mixture model with underlying kernel function as defined in Equation 9.28 [15, 143].

The entropy of a Gaussian Mixture Model is defined as

$$H_G = \frac{c}{2} + \frac{c}{2}\log(2\pi) + \frac{1}{2}\log(|\boldsymbol{\Sigma}|). \tag{9.29}$$

To obtain a measure of uncertainty of our model, for a given image we first normalize the entropy values over all pose hypotheses individually for both rotation and translation estimates, and finally obtain a measure of (un)certainty as the sum of both rotational and translational normalized entropy.

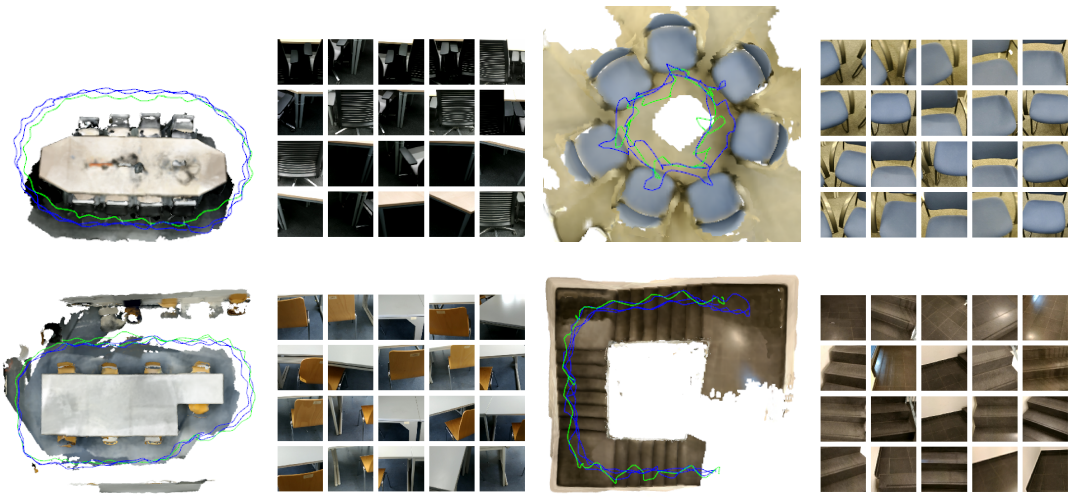


Fig. 9.7. Ground truth training (blue) and test (green) camera trajectories of our real ambiguous scenes and example RGB images.

On the output we now have *fourteen* quantities per distribution density: seven for the Bingham and additional three for translation as well as three for variances of the multivariate Gaussian. In total as we have K mixture components resulting in $K \times 14$ output entities. Our architecture is shown in Figure 9.6. Once again, during training we apply the MHP scheme explained above to avoid mode collapse and diversify the predictions. In practice, we first train the network to predict the translation and its variance. Then, intuitively, recovering the associated rotation should be an easier task, after which we fine-tune the network on all components of the distribution.

Network and training details We resize the input images to a height of 256 pixels and use random crops of size 224×224 for training. For testing we use the central crop of the image. As described in the main paper we use a ResNet-34 [87] as our backbone network, which was pre-trained on ImageNet [184], and remove the final classification layers. Fully-connected layers are then appended, where we output K camera pose hypotheses, $\mathbf{q}$ and $\mathbf{t}$, corresponding distribution parameters, $\mathbf{\Lambda}$ and $\mathbf{\Sigma}$, as well as shared mixture weights $\pi(\mathbf{X}, \mathbf{\Gamma})$. In case of our single component and Bingham-MDN models we use a softmax activation function, such that $\sum_{j=1}^{K} \pi_j(\mathbf{X}, \mathbf{\Gamma}) = 1$ holds true. In our MHP version, we first apply a ReLU activation function, that, during training, is passed to a cross-entropy loss function. Once trained, we again apply a softmax on the final weights to form a valid mixture model. If not stated otherwise we use $K = 50$ hypotheses for our multimodal models as well as the baselines.

9.7.1 Datasets

We evaluate on the standard datasets of 7-Scenes [199] and Cambridge Landmarks [107], which are widely used to analyze image-based localization methods. As introduced in the previous parts of this thesis the 7-Scenes dataset from Microsoft contains seven indoor scenes. The Cambridge Landmarks dataset on the other hand contains five outdoor scenes located around the Cambridge Univerity. This dataset provides RGB images for training as well as test data, captured using a mobile phone and ground truth camera pose computed using structure

Tab. 9.1. Spatial extent of our newly created ambiguous scenes dataset in meter.

Blue Chairs	Meeting Table	Seminar	Staircase	Staircase Ext.
$5 \times 4.6 \times 1.3$	$4.3 \times 5.8 \times 1.4$	$5.3 \times 7.8 \times 2.6$	$4.9 \times 4.4 \times 5.1$	$5.6 \times 5.2 \times 16.6$

from motion. In addition to these two datasets, we created synthetic as well as real datasets, that are specifically designed to contain repetitive structures and allow us to assess the real benefits of our approach. For synthetic data we render models from 3DWarehouse[1] and create camera trajectories, e.g. a circular movement around the object, such that ambiguous views are ensured to be included in our dataset. Specifically we use a *dining table* and a *round table* model with discrete modes of ambiguities. In addition, we create highly ambiguous real scenes using Google Tango and the graph-based SLAM approach RTAB-Map [117]. We acquire RGB and depth images as well as distinct ground truth camera trajectories for training and testing. We also reconstruct those scenes. However, note that only the RGB images and corresponding camera poses are required to train our model and the reconstructions are used for visualization only. Figure 9.7 shows ground truth training and testing camera trajectories, plotted with Open3D [245], as well as example batch images we acquired for our ambiguous scene dataset. In total our training and test sets consist of 2414 and 1326 frames, respectively. The spatial extent of our scenes can be found in Table 9.1.

9.7.2 Baselines

We compare our approach to current state-of-the-art direct camera pose regression methods, PoseNet [105] and MapNet [26], that output a single pose prediction. More importantly, we assess our performance against two state-of-the-art approaches, namely Bayesian PoseNet [106] and VidLoc [50], that are most related to our work and predict a distribution over the pose space by using dropout and mixture density networks, respectively. We further include the *unimodal* predictions as well as BMMs trained using mixture density networks [15, 73] as baselines. We coin the latter Bingham-MDN or in short *BMDN*. In addition we compare between explicitly learning the mixture coefficient, *Ours-RWTA* (see Equation 9.24), and implicit learning, *Ours-BMDN+RWTA* (see Equation 9.20). For better understanding, we summarize the different versions of our models used in the latter evaluations in Table 9.2, to show the individual aspects of each model.

9.7.3 Experiments and Results

To evaluate our method we consider two cases: (1) camera relocalization in non-ambiguous scenes, where our aim is to not only predict the camera pose, but the posterior of both rotation and translation that can be used to associate each pose with a measure of uncertainty; (2) we create a highly ambiguous environment, where similar looking images are captured from very different viewpoints. We show the problems current regression methods suffer from in handling such scenarios and in contrast show the merit of our proposed method.

[1] https://3dwarehouse.sketchup.com/

Tab. 9.2. Summary of the provided information by the baseline and our methods. Direct regression, such as PoseNet, does not provide uncertainty information or multiple hypotheses, whereas MC-Dropout only includes per pose uncertainty. Except for our unimodal model, our model provides multiple hypotheses as well as per hypothesis uncertainty estimation.

	PoseNet [107]	Bayesian PoseNet [106]	Unimodal	Bingham-MDN	Ours-RWTA	Ours-MBDN-RWTA
Uncertainty	-	(√)	√	√	√	√
Multiple Hypotheses	-	√	-	√	√	√

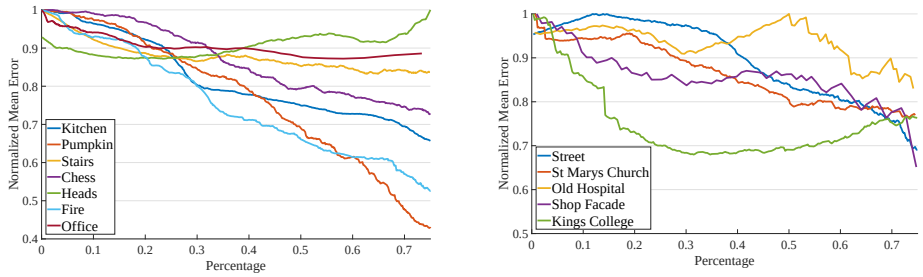
Tab. 9.3. Evaluation in non-ambiguous scenes, displayed is the median rotation and translation error on the 7-Scenes dataset.

Dataset	7-Scenes						
[° / m]	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs
PoseNet [105]	4.48 / 0.13	**11.3** / 0.27	13.0 / 0.17	5.55 / 0.19	4.75 / 0.26	5.35 / 0.23	12.4 / 0.35
MapNet [26]	**3.25** / **0.08**	11.69 / 0.27	13.2 / 0.18	**5.15** / **0.17**	**4.02** / **0.22**	**4.93** / 0.23	12.08 / 0.3
Bayes-PoseNet [106]	7.24 / 0.37	13.7 / 0.43	**12.0** / 0.31	8.04 / 0.48	7.08 / 0.61	7.54 / 0.58	13.1/ 0.48-
VidLoc [50]	- / 0.18	- / **0.26**	- / 0.14	- / 0.26	- / 0.36	- / 0.31	- / **0.26**
Ours-Unimodal	4.97 / 0.1	12.87 / 0.27	14.05 / **0.12**	7.52 / 0.2	7.11 / 0.23	8.25 / **0.19**	13.1 / 0.28
Ours-Bingham-MDN	4.35 / 0.1	11.86 / 0.28	12.76 / **0.12**	6.55 / 0.19	6.9 / **0.22**	8.08 / 0.21	**9.98** / 0.31

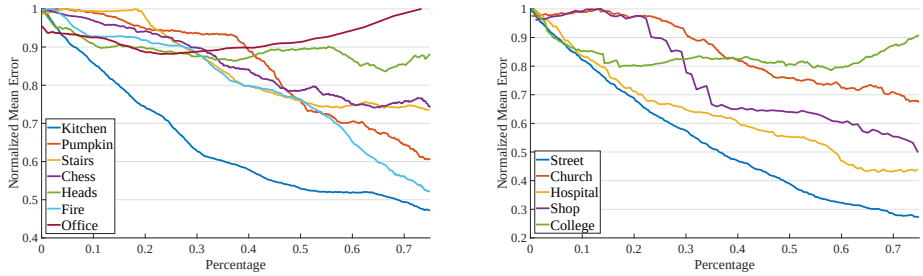
Error Metrics Note that, under ambiguities a best mode is unlikely to exist. In those cases, as long as we can generate a hypothesis that is close to the ground truth, our network is considered successful. For this reason, in addition to the weighted mode, which we pick according to the predicted mixture coefficients, we will also speak of the so called *Oracle* Error, assuming an oracle that is able to choose the best of all predictions: the one closest to the ground truth. In addition, we report the *Self-EMD* (SEMD) [133], the earth movers distance [180] of turning a multi-modal distribution into a unimodal one. With this measure we can evaluate the diversity of predictions, where the unimodal distribution is chosen as the predicted mode of the corresponding method. Note that this measure by itself does not give any indication about the accuracy of the prediction.

Tab. 9.4. Evaluation in non-ambiguous scenes, displayed is the median rotation and translation error on the Cambridge Landmarks dataset, where numbers for MapNet are taken from [193].

Dataset	Cambridge Landmarks				
[° / m]	Kings College	Old Hospital	Shop Facade	St. Marys Church	Street
PoseNet [105]	**1.04** / 0.88	**3.29** / 3.2	**3.78** / 0.88	**3.32** / 1.57	25.5 / 20.3
MapNet [26]	1.89 / 1.07	3.91 / 1.94	4.22 / 1.49	4.53 / 2.0 /	-
Bayes-PoseNet [106]	4.06 / 1.74	5.12 / 2.57	7.54 / 1.25	8.38 / 2.11	-
Ours-Unimodal	1.77 / 0.88	3.71 / **1.93**	4.74 / **0.8**	6.19 / 1.84	**24.08** / 16.8
Ours-Bingham-MDN	2.08 / **0.83**	3.64 / 2.16	4.93 / 0.92	6.03 / **1.37**	36.88 / **9.69**



(a) Rotation Uncertainty



(b) Translation Uncertainty

Fig. 9.8. Uncertainty evaluation on the 7-Scenes and Cambridge Landmarks datasets, showing the correlation between predicted uncertainty and pose error. Based on the entropy of our predicted distribution uncertain samples are gradually removed. We observe that as we remove the uncertain samples the overall error drops indicating a strong correlation between our predictions and the actual erroneous estimations.

Evaluation in non-ambiguous scenes

We first evaluate our method on the publicly available 7-Scenes [199] and Cambridge Landmarks [107] datasets. As most of the scenes contained in these datasets do not show highly ambiguous environments, we consider them to be non-ambiguous, although, obviously we can not guarantee that some ambiguous views can arise in these datasets as well, such as in the *Stairs* scene of the 7-Scenes dataset. Both datasets have extensively been used to evaluate camera pose estimation methods. Thus, we compare to current state-of-the-art direct camera pose regression methods, PoseNet [105] and MapNet [26], that output a single pose prediction. More importantly, we assess our performance against two state-of-the-art approaches, namely BayesianPoseNet [106] and VidLoc [50], that are most related to our work and predict a distribution over the pose space by using dropout and mixture density networks, respectively. Following the state of the art, we report the median rotation and translation errors, the results of which can be found in Tables 9.3 and 9.4. In comparison to methods that output a single pose prediction, PoseNet [105] and MapNet [26], our methods achieve similar results, whereas, especially in translation our method outperforms uncertainty methods, namely BayesianPoseNet [106] and VidLoc [50], on most scenes.

Uncertainty evaluation One benefit of our method is that we can use the resulting variance of the predicted distribution as a measure of uncertainty in our predictions. The resulting correlation between pose error and uncertainty can be seen in Figure 9.8, where we gradually remove the most uncertain predictions and plot the mean error for the remaining samples. The strong inverse correlation between the actual errors versus our confidence shows that whenever our algorithm labels a prediction as uncertain it is also likely to be a bad estimate.

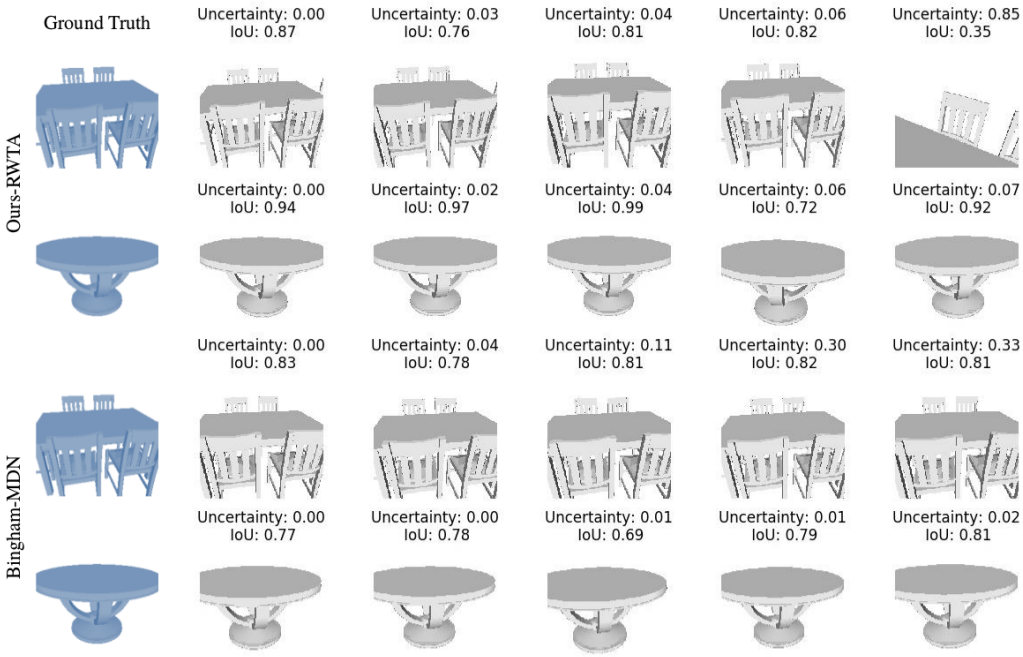


Fig. 9.9. Renderings of the top five camera pose hypotheses according to their uncertainty values for our Bingham-MDN and MHP version, Ours-RWTA. Further we show the corresponding ground truth query images as well as the intersection over union of the ground truth and predicted renderings.

It has been shown that current direct camera pose regression methods still have difficulties in generalizing to views that differ significantly from the camera trajectories seen during training [193]. In addition these methods face further problems when deployed in a highly ambiguous environment. For this purpose, we analyze the performance of direct regression methods when presented with ambiguous views. In this scenario even similar trajectories can confuse the network and easily lead to wrong predictions, for which our method proposes a solution.

Evaluation in ambiguous scenes

We start with quantitative evaluations on our synthetic as well as real scenes before showing qualitative results of our and the baseline methods.

Quantitative evaluations Due to the design of our synthetic table scenes, we know that there are two or four possible modes for each image in *dining* and *round* table scenes respectively. Hence, we analyze the mode predictions of our model by computing the accuracy of correctly detected modes of the true posterior distribution. A mode is considered as found if there exists one pose hypothesis that falls into a certain rotational and translational threshold of it. For a threshold of 5° and 10% of the ground truth camera trajectory's circle's diameter in translation, MC-Dropout obtains an accuracy of 50%, finding one mode for each image, whereas the accuracy of Ours-RWTA on average achieves 96% on our *dining table* scene. We include additional results on our second synthetically created scene, a *round table*. By construction of this scene, we would expect four modes to exist and be found by our model. On average our model shows a detection rate of 99.1%, in comparison to 24.8% of MC-Dropout. Furthermore, we render the models from the predicted poses and compute the intersection over union (IoU) with the ground truth renderings in Figure 9.9. Considering the hypothesis with the highest

Tab. 9.5. Ratio of correct poses for several thresholds, evaluated on our ambiguous scenes dataset.

	Threshold	PoseNet [107]	Unimodal	Bingham-MDN	MC-Dropout [106]	Ours-RWTA	MC-Dropout Oracle	Ours-RWTA Oracle
Blue Chairs	10° / 0.1m	0.19	0.29	0.24	**0.39**	0.35	0.40	**0.58**
	15° / 0.2m	0.69	0.73	0.75	0.78	**0.81**	0.90	**0.94**
	20° / 0.3m	**0.90**	0.86	0.80	0.88	0.82	0.95	**1.00**
Meeting Table	10° / 0.1m	0.0	0.02	0.01	0.04	**0.05**	**0.13**	0.12
	15° / 0.2m	0.05	0.12	0.07	0.13	**0.28**	0.27	**0.56**
	20° / 0.3m	0.10	0.19	0.10	0.22	**0.39**	0.32	**0.78**
Staircase	10° / 0.1m	0.14	0.11	0.04	0.13	**0.18**	**0.27**	0.19
	15° / 0.2m	0.45	0.48	0.15	0.32	**0.50**	**0.54**	0.53
	20° / 0.3m	0.60	0.62	0.25	0.49	**0.68**	0.70	**0.74**
Staircase Extended	10° / 0.1m	0.07	0.06	0.06	0.02	**0.09**	**0.16**	0.09
	15° / 0.2m	0.31	0.26	0.21	0.14	**0.39**	**0.45**	0.40
	20° / 0.3m	0.49	0.41	0.32	0.31	**0.58**	**0.64**	**0.64**
Seminar Room	10° / 0.1m	**0.37**	0.11	0.06	0.18	0.35	**0.46**	0.36
	15° / 0.2m	0.81	0.36	0.23	0.57	**0.83**	**0.85**	0.83
	20° / 0.3m	0.90	0.57	0.40	0.78	**0.95**	0.90	**0.95**
Average	10° / 0.1m	0.15	0.12	0.08	0.15	**0.20**	**0.28**	0.27
	15° / 0.2m	0.46	0.39	0.28	0.39	**0.56**	0.60	**0.65**
	20° / 0.3m	0.60	0.53	0.37	0.54	**0.68**	0.70	**0.82**

Tab. 9.6. Ratio of correctly detected modes for various translational thresholds. The threshold for rotation is set to 15.0 °.

		Threshold			
Scene	Method	0.1m	0.2m	0.3m	0.4m
Blue Chairs	MC-Dropout	0.11	0.15	0.16	0.16
	Ours-RWTA	**0.36**	**0.79**	**0.80**	**0.80**
Meeting Table	MC-Dropout	0.04	0.07	0.09	0.11
	Ours-RWTA	**0.10**	**0.43**	**0.63**	**0.73**

weight, on average our Bingham-MDN reaches 0.62, whereas our MHP distribution model, *Ours-RWTA*, achieves an intersection over union of 0.88.

On our real scenes, we report the ratio of correct poses, where a pose is considered to be correct if both the rotation and translation errors are below a pre-defined threshold. Table 9.5 shows the accuracy of our baseline methods in comparison to ours for various thresholds. Especially on our *meeting table* scene, it can be seen that the performance of direct camera pose regression methods that suffer from mode collapse significantly drops due to the presence of ambiguities in the scene. As a result of our diverse mode predictions of Ours-RWTA, which is indicated by the high Oracle accuracy, we are able to improve upon our baselines predictions.

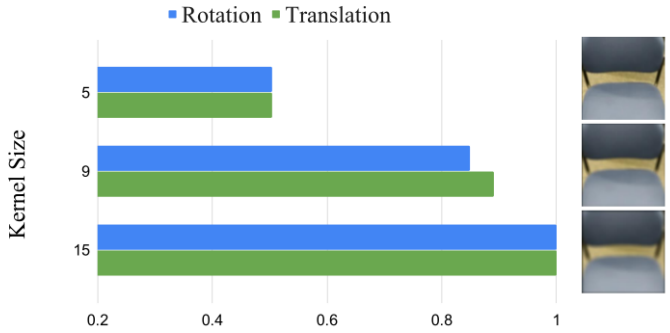


Fig. 9.10. Change in uncertainty prediction in the presence of increasing image blur. For varying kernel sizes of a Gaussian filter used to blur the input images, we compute the average uncertainty over all images obtained from the predictions of our model. Reported here are the normalized values.

For our real scenes, we obtain a ground truth estimate by training an autoencoder on reconstructing the input images and using the resulting feature descriptors to obtain the nearest neighbor camera poses. Then we cluster the resulting camera poses using a Riemannian Mean Shift algorithm [207] and use the centroids of the resulting clusters as "ground truth" modes. We visually verify the results. Table 9.6 shows the percentage of correctly detected modes for our method in comparison to MC-Dropout. The results support our qualitative observations that MC-Dropout, suffers from mode collapse such that even with increasing threshold the number of detected modes does not increase significantly.

Uncertainty evaluation Due to fast camera movements, motion blur easily arises in camera localization applications and is one factor that can lead to poor localization performance. As a first step in handling such problems, additional information in the form of uncertainty predictions could aid in detecting such events. Therefore, to evaluate how our model performs in the presence of noise, we use our single component model, i.e. $K = 1$, trained on the original input images, and blur the test RGB images to evaluate the change in uncertainty prediction of the model. Ideally, with increasing image blur, we would expect our model to be less certain in its predictions. For this purpose, we apply a Gaussian filter to the input images, with varying kernel sizes, and report the change in uncertainty prediction in Figure 9.10 on the blurred images. We use the entropy over each image to obtain a measure of uncertainty and compute the mean over our dataset images. For visualization, we show the normalized values. An increase in uncertainty could be clearly observed with growing kernel size and thus highly blurred images.

Qualitative evaluations Qualitative results of our proposed model on our synthetic *dining table* dataset are shown in Figure 9.11 and 9.12. MC-Dropout as well as our finite mixture model, *Bingham-MDN*, suffer from mode collapse. In comparison, the proposed MHP model is able to capture plausible, but diverse, modes as well as associated uncertainties. In contrast to other methods that obtain an uncertainty value for one prediction, we obtain uncertainty values for each hypothesis. This way, we could easily remove non-meaningful predictions, that for example can arise in the WTA and RWTA training schemes. Resulting predicted Bingham distributions are further visualized in Figure 9.13. Furthermore, we render the objects from the predicted camera poses of our models in Figure 9.9. There, we show the most certain predictions sorted according to the entropy of the resulting Bingham and Gaussian

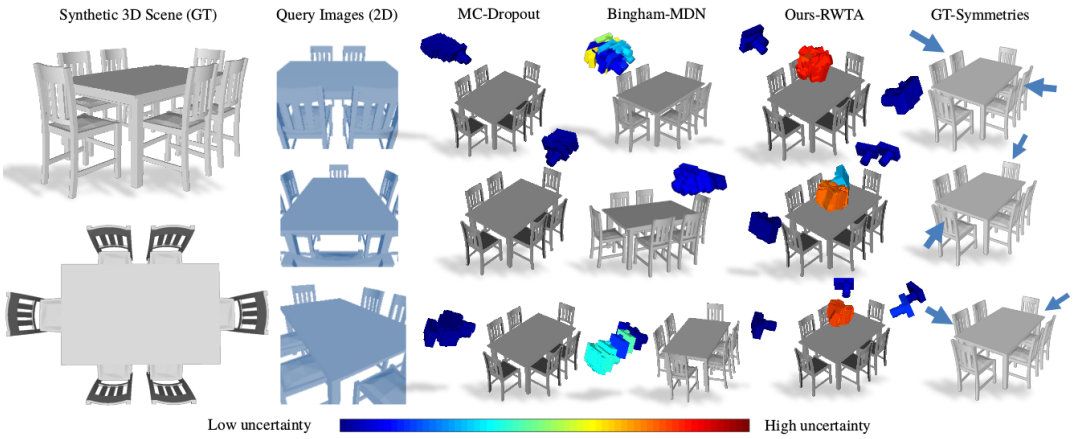


Fig. 9.11. Qualitative results on our synthetic *dining table* dataset. Camera poses are colored according to their uncertainty. Viewpoints are adjusted for best perception.

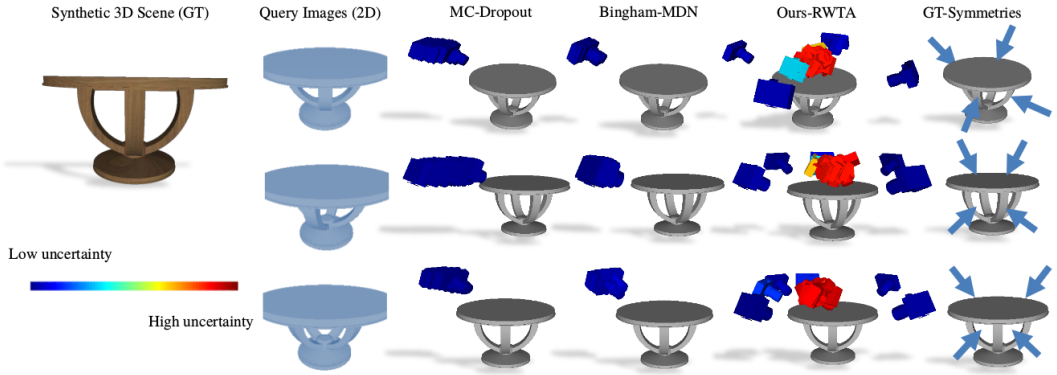


Fig. 9.12. Additional qualitative results of our synthetically created *round table* dataset. If available, camera poses are colored by their uncertainty.

distributions. Figure 9.14 shows qualitative results on our ambiguous real scenes dataset. Again, MC-Dropout and Bingham-MDN suffer from mode collapse. More importantly, these methods are unable to predict reasonable poses given highly ambiguous query images. That is most profound in our *Meeting Table* scene, where the predicted camera poses fall on the opposite side of the ground truth one. Additionally we show a qualitative evaluation of our predictions in our largest scene, *Staircase Extended*, in Figure 9.15.

Ablation Studies

To evaluate or method we conduct further ablative studies. We begin with analyzing different variations on how to effectively optimize a multiple hypotheses network and evaluate recent network architectures proposed in the literature as our backbone. We then extend our model and evaluate our proposed implicit learning of mixture coefficients. Further we show the effect of distance function in multiple hypothesis training to choose the best branches for learning during training. Then, we compare various mathematical methods on how to construct the Bingham distribution from a neural network's prediction. Further we show the effect of different rotation parameterizations when used in a mixture density network for all of our and the baseline methods.

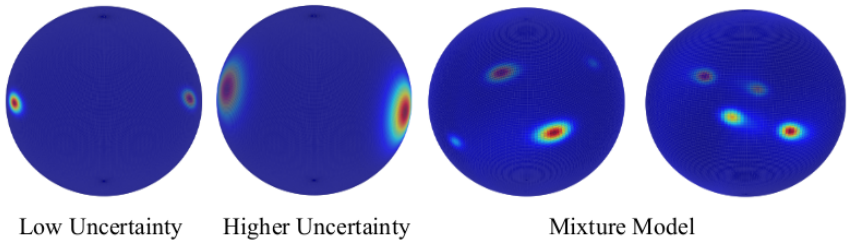


Fig. 9.13. Bingham distributions plotted on the unit sphere. Single model predictions with low uncertainty, higher uncertainty and the mixture model of Ours-RWTA evaluated on the *Blue Chairs* scene of our ambiguous real scenes dataset.

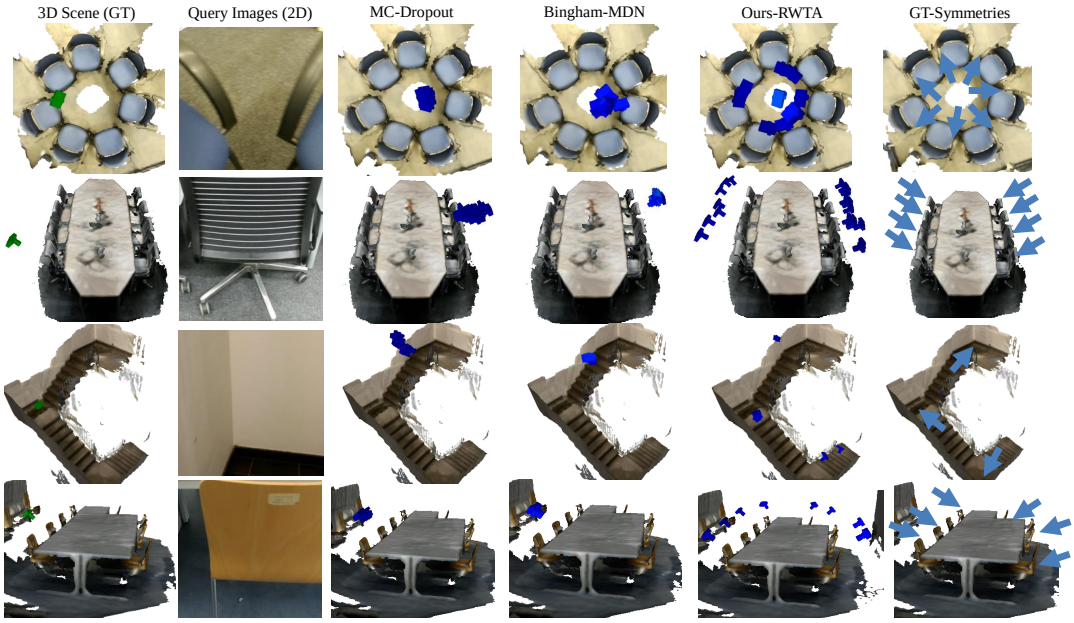


Fig. 9.14. Qualitative results in our ambiguous dataset. For better visualization, if available, camera poses have been pruned by their uncertainty values.

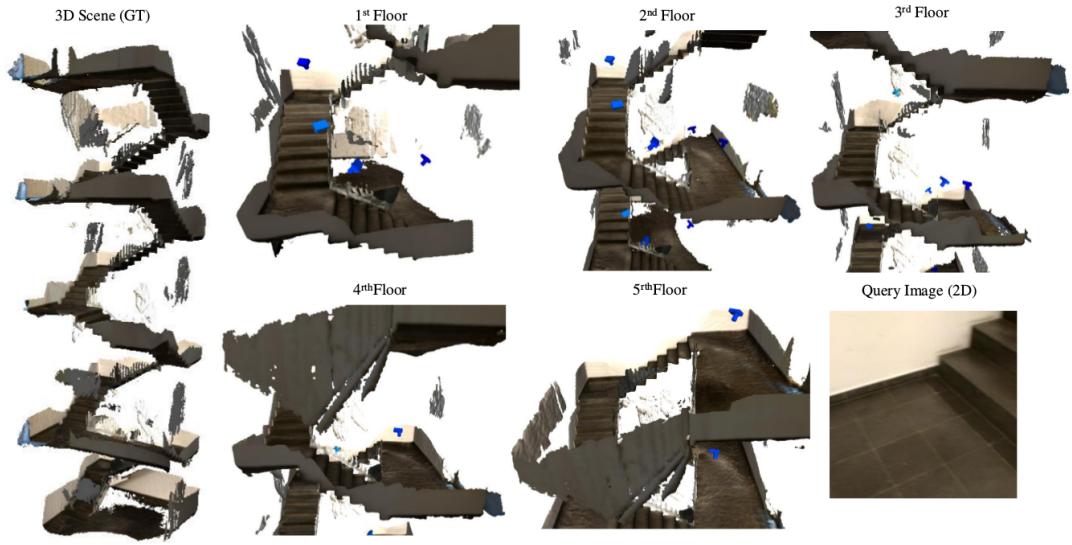


Fig. 9.15. Qualitative results of our model on the ambiguous scenes dataset, *Staircase Extended* scene.

Tab. 9.7. Comparison between different MHP variants, RWTA [182] and EWTA [133], and Bingham-MDN, averaged over all scenes.

Threshold	Bingham-MDN	EWTA(k=50)	EWTA (k=25)	RWTA (k=1, used)
10° / 0.1m	0.08	0.12	0.18	0.20
15° / 0.2m	0.28	0.34	0.40	0.56
20° / 0.3m	0.37	0.47	0.51	0.68

Tab. 9.8. Averaged ratio of correct poses for different backbone networks over all scenes of our real ambiguous scenes dataset.

	Threshold	PoseNet	Unimodal	Bingham-MDN	MC-Dropout	Ours-RWTA
ResNet-34	10° / 0.1m	0.15	0.12	0.08	0.15	**0.20**
	15° / 0.2m	0.46	0.39	0.28	0.39	**0.56**
	20° / 0.3m	0.60	0.53	0.37	0.54	**0.68**
ResNet-18	10° / 0.1m	0.15	0.16	0.09	0.15	**0.19**
	15° / 0.2m	0.47	0.42	0.29	0.39	**0.52**
	20° / 0.3m	0.60	0.54	0.39	0.54	**0.66**
ResNet-50	10° / 0.1m	**0.20**	0.15	0.10	0.15	**0.20**
	15° / 0.2m	0.49	0.36	0.30	0.40	**0.55**
	20° / 0.3m	0.62	0.53	0.38	0.53	**0.69**
Inception-v3	10° / 0.1m	0.11	0.10	0.11	0.08	**0.18**
	15° / 0.2m	0.38	0.33	0.38	0.31	**0.49**
	20° / 0.3m	0.55	0.53	0.52	0.49	**0.63**

Multiple hypothesis estimation Recently, [133] have proposed EWTA, an evolving version of WTA, to alleviate the problems arising with the original MHP training schemes proposed in [182]. Updating the top k hypotheses instead of only the best one, EWTA increases the number of hypotheses that are actually used during training. This results in fewer wrong mode predictions that do not match the actual distribution. However, in our case we have found wrong predictions to have very high uncertainty so that, if desired, they can easily be filtered. Nevertheless, we evaluated the different versions of MHP training schemes for our particular application for which the results can be found in Table 9.7. As it is not straightforward how k should be chosen, we 1) start with $k = K$, where K is the number of hypotheses and gradually decrease k until $k = 1$ (as proposed in [133]) and 2) start with the best half hypotheses, i.e. $k = 0.5 \cdot K$. We set $K = 50$ in our experiments. We have found this parameter to strongly influence the accuracy of our model. Therefore, and since wrong predictions can be easily be identified with our model, we chose to remain with the original version of RWTA to train our models.

Backbone network To evaluate the effect of different network architectures on our model, we change the backbone network of ours and the state-of-the-art baseline methods. Namely,

Tab. 9.9. Inference time of our method, *Ours-RWTA*, with respect to the number of hypothesis.

PoseNet	$K = 1$	$K = 50$	$K = 200$	$K = 500$
7.23ms	7.27ms	8.11ms	8.19ms	8.74ms

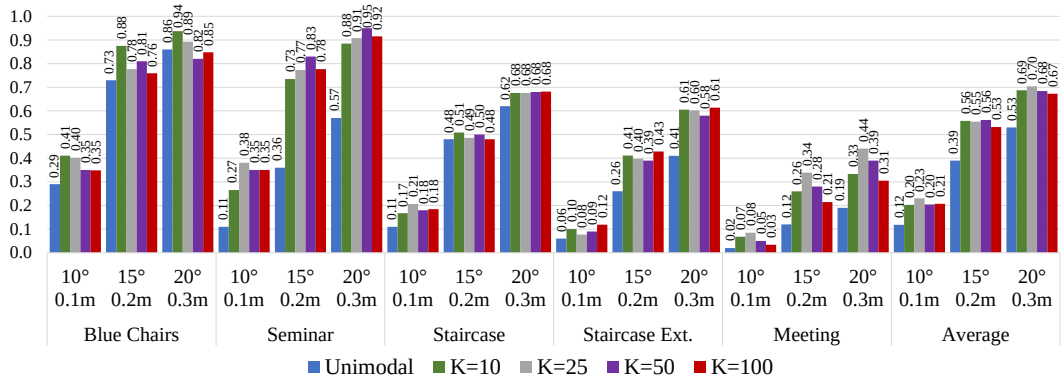


Fig. 9.16. Influence of the number of hypotheses, i.e. parameter K, on the performance of our method, Ours-RWTA. We present the ratio of correctly predicted poses under varying parameter values of K.

we compare between resnet variants ResNet-18, ResNet-34 and ResNet-50 and Inception-v3. Most recent state-of-the-art image based localization methods [6, 26, 165] use a version of ResNet. All networks remain initialized from a on ImageNet pre-trained model. We report our findings in Table 9.8. Naturally all methods are, to some extent, dependant on the features that serve as input to the final pose regression layers. However, it seems MC-Dropout and Ours-RWTA are less affected by the change in feature representation resulting from different network architectures.

Number of Hypotheses and Computational Times Incorporating our method into an existing regression model, simply leads to a change in the last fully-connected layers of the network. We extend the last layer to output an additional $(K-1)\cdot 4$ and $(K-1)\cdot 3$ parameters for predicting the camera pose, as well as overall $6\cdot K$ for uncertainty prediction of both rotation and translation. Further, we incorporate three extra layers for the mixture coefficients. We run our model on a 8GB NVIDIA GeForce GTX 1080 graphics card and report the inference time of our network with respect to K in Table 9.9. In comparison to a direct regression method our model with $K = 50$ incurs a negligible computational overhead around $1ms$.

Further, we evaluate the effect of hyper-parameter K, i.e. the number of hypothesis to be regressed, for our proposed method. Based on the results, which are summarized in Figure 9.16, we suspect the optimal number of hypotheses to be dependent on the spatial extent of the scene and on the ambiguities contained in them. However, due to the increased complexity of the model as well as instability issues during training, we observed a drop in performance with high increase of the number of hypotheses.

Implicit learning of mixture coefficients We now evaluate our extended method, Ours-MBDN+RWTA (see 9.20), that allows for implicit learning of the mixture coefficients without succumbing to the pitfalls of ordinary mixture density networks such as mode collapse. Table

9.10 shows the accuracy of our baseline methods in comparison to ours for various thresholds. We are able to improve upon our baseline model, Ours-RWTA, in overall performance, as well as in SEMD as reported in Table 9.11. Further, our model is able to provide diverse predictions and capture multiple modes, which is indicated by the high Oracle accuracy, see Table 9.10.

Tab. 9.10. Ratio of correct poses on our ambiguous scenes for several thresholds. We report the results of our Ours-RWTA and Ours-MBDN+RWTA for several number of hypotheses K.

	Threshold	Ours-RWTA	Ours-MBDN+RWTA				Ours-RWTA Oracle	Ours-MBDN+RWTA Oracle			
Num. Hypotheses		50	5	10	25	50	50	5	10	25	50
Blue Chairs (A)	10° / 0.1m	0.35	0.40	**0.48**	0.35	0.39	0.58	0.51	0.54	0.46	0.41
	15° / 0.2m	0.81	0.85	**0.92**	0.80	0.79	0.94	0.87	0.91	0.92	0.92
	20° / 0.3m	0.82	0.89	**0.96**	0.87	0.85	1.0	0.94	0.96	0.97	0.99
Meeting Table (B)	10° / 0.1m	0.05	0.03	0.07	**0.08**	0.03	0.12	0.06	0.08	0.12	0.15
	15° / 0.2m	0.28	0.26	**0.33**	0.31	0.32	0.56	0.29	0.42	0.57	0.58
	20° / 0.3m	0.39	0.34	**0.42**	0.38	0.41	0.78	0.41	0.55	0.73	0.81
Staircase (C)	10° / 0.1m	0.18	**0.20**	0.18	0.18	0.17	0.19	0.20	0.23	0.21	0.23
	15° / 0.2m	**0.50**	0.47	0.47	0.47	0.49	0.53	0.49	0.47	0.51	0.58
	20° / 0.3m	**0.68**	0.64	0.66	0.64	0.64	0.74	0.64	0.71	0.71	0.76
Staircase Extended (D)	10° / 0.1m	0.09	0.10	0.06	0.09	**0.11**	0.09	0.12	0.08	0.10	0.12
	15° / 0.2m	0.39	0.43	0.38	0.44	**0.46**	0.40	0.41	0.38	0.47	0.47
	20° / 0.3m	0.58	0.59	0.60	0.62	**0.64**	0.64	0.56	0.60	0.66	0.69
Seminar Room (E)	10° / 0.1m	0.35	**0.39**	0.38	0.35	0.37	0.36	0.41	0.42	0.43	0.41
	15° / 0.2m	0.83	0.77	0.78	**0.84**	0.80	0.83	0.77	0.80	0.88	0.82
	20° / 0.3m	**0.95**	0.88	0.93	0.94	0.92	0.95	0.89	0.93	0.95	0.94
Average	10° / 0.1m	0.20	0.23	**0.24**	0.21	0.22	0.27	0.26	0.27	0.27	0.26
	15° / 0.2m	0.56	0.56	**0.58**	0.57	0.57	0.65	0.56	0.60	0.67	0.67
	20° / 0.3m	0.68	0.67	**0.71**	0.69	0.69	0.82	0.69	0.75	0.80	0.84

The distance function in MHP In addition we provide ablation studies and results on the chosen distance function, ℓ_1 or negative log likelihood, for choosing the best branch in our MHP training schemes. We report the results in Table 9.12. Overall we have found ℓ_1 to outperform the likelihood version, which we found to be mostly due to the rotation and translational difference in the log likelihood. ℓ_1 seems to be less affected.

Tab. 9.11. SEMD of our methods indicating highly diverse predictions.

Method/Scene	Blue Chairs	Meeting Table	Staircase	Staircase Ext.	Seminar Room
MC-Dropout	0.06	0.11	0.13	0.26	0.1
Ours-RWTA	1.19	2.13	2.04	3.81	1.70
Ours-MBDN+RWTA	**1.20**	**2.53**	**2.24**	**4.35**	**2.22**

Tab. 9.12. Influence for different distance functions on the MHP training scheme. We report the average recall over all scenes in our ambiguous scene dataset for different thresholds.

Threshold	Ours-RWTA (l_1)	Ours-MBDN+RWTA (l_1)	Ours-MBDN+RWTA (logp)
10° / 0.1m	0.20	**0.22**	0.19
15° / 0.2m	0.56	**0.57**	0.53
20° / 0.3m	0.68	**0.69**	0.66

Tab. 9.13. Ratio of correct poses for several thresholds of Gram-Schmidt (*G*), Skew-Symmetric (*S*) and Birdal et al.(B) methods to construct **V**.

G / S / B	Threshold	Unimodal	Bingham-MDN	Ours-RWTA	Ours-MBDN+RWTA
Blue Chairs (A)	10° / 0.1m	0.24 / 0.23 / 0.29	0.04 / 0.17 / 0.24	0.30 / 0.12 / 0.35	0.46 / 0.29 / 0.39
	15° / 0.2m	0.63 / 0.58 / 0.73	0.15 / 0.49 / 0.75	0.73 / 0.39 / 0.81	0.82 / 0.66 / 0.79
	20° / 0.3m	0.76 / 0.73 / 0.86	0.18 / 0.59 / 0.80	0.79 / 0.43 / 0.82	0.86 / 0.71 / 0.85
Meeting Table (B)	10° / 0.1m	0.02 / 0.07 / 0.02	0.04 / 0.01 / 0.01	0.04 / 0.09 / 0.05	0.07 / 0.09 / 0.03
	15° / 0.2m	0.16 / 0.20 / 0.12	0.18 / 0.14 / 0.07	0.12 / 0.23 / 0.28	0.30 / 0.26 / 0.32
	20° / 0.3m	0.24 / 0.25 / 0.19	0.21 / 0.24 / 0.10	0.18 / 0.27 / 0.39	0.39 / 0.33 / 0.41
Staircase (C)	10° / 0.1m	0.17 / 0.16 / 0.11	0.21 / 0.16 / 0.04	0.17 / 0.14 / 0.18	0.24 / 0.18 / 0.17
	15° / 0.2m	0.46 / 0.51 / 0.62	0.43 / 0.37 / 0.15	0.46 / 0.42 / 0.50	0.52 / 0.47 / 0.49
	20° / 0.3m	0.62 / 0.64 / 0.62	0.60 / 0.49 / 0.25	0.60 / 0.62 / 0.68	0.63 / 0.62 / 0.64
Staircase Extended (D)	10° / 0.1m	0.04 / 0.04 / 0.06	0.04 / 0.07 / 0.06	0.05 / 0.06 / 0.09	0.08 / 0.05 / 0.11
	15° / 0.2m	0.16 / 0.16 / 0.26	0.19 / 0.29 / 0.21	0.23 / 0.26 / 0.39	0.41 / 0.28 / 0.46
	20° / 0.3m	0.27 / 0.27 / 0.41	0.31 / 0.41 / 0.32	0.34 / 0.36 / 0.58	0.61 / 0.38 / 0.64
Seminar Room (E)	10° / 0.1m	0.27 / 0.33 / 0.06	0.30 / 0.35 / 0.06	0.15 / 0.28 / 0.35	0.19 / 0.30 / 0.37
	15° / 0.2m	0.69 / 0.69 / 0.23	0.56 / 0.59 / 0.23	0.47 / 0.70 / 0.83	0.52 / 0.63 / 0.80
	20° / 0.3m	0.82 / 0.80 / 0.40	0.64 / 0.70 / 0.40	0.58 / 0.79 / 0.95	0.63 / 0.77 / 0.92
Average	10° / 0.1m	0.15 / **0.16** / 0.11	**0.13** / **0.13** / 0.08	0.14 / 0.14 / **0.20**	0.21 / 0.18 / 0.22
	15° / 0.2m	0.42 / **0.43** / 0.36	0.30 / **0.38** / 0.28	0.40 / 0.40 / **0.56**	0.51 / 0.46 / 0.57
	20° / 0.3m	**0.54** / **0.54** / 0.50	0.39 / **0.49** / 0.37	0.50 / 0.49 / **0.68**	0.63 / 0.56 / 0.69

Constructing V There are multiple ways in which we can construct the orthonormal matrix **V**. In particular we compare between the method of Birdal et al., Gram Schmidt orthonormalization and the construction using skew-symmetric matrices, for which the results can be found in Table 9.13. In comparison to Gram Schmidt orthonormalization by using the remaining methods only four parameters have to be estimated instead of the 16 entries of the matrix **V**. For our unimodal as well as multimodal MDN we found the Gram-Schmidt and Skew-Symmetric construction to outperform Birdal's. However, for our method, Ours-RWTA, the latter performs best and additionally achieves overall the best performance in comparison the the remaining methods and constructions.

Rotation parameterization When training deep learning models, the choice of rotation parameterization has posed an ongoing debate. PoseNet [107] proposes to use quaternions as normalization can be easily done during training and the ambiguities they pose resolved by mapping them to one hemisphere. MapNet [26] further shows improvements in using the axis angle representation. Recently it has been shown that any representation with four or

Tab. 9.14. Ratio of correct poses when using the continuous 6d representation of [247] in order to model rotations instead of a Bingham distribution on the quaternion.

	Threshold	Geo + ℓ_1	Unimodal	MDN	MC-Dropout	9D-Ours-RWTA	9D-Ours-MBDN+RWTA
Blue Chairs	10° / 0.1m	0.41	**0.48**	0.01	0.26	0.38	0.38
	15° / 0.2m	**0.90**	0.89	0.14	0.83	0.81	0.79
	20° / 0.3m	**0.96**	0.92	0.23	0.91	0.84	0.81
Meeting Table	10° / 0.1m	0.03	0.03	0.02	0.02	0.06	**0.07**
	15° / 0.2m	0.16	0.16	0.11	0.13	0.29	**0.33**
	20° / 0.3m	0.22	0.23	0.14	0.21	0.38	**0.42**
Staircase	10° / 0.1m	0.17	0.19	0.12	0.12	0.18	**0.20**
	15° / 0.2m	0.46	**0.51**	0.36	0.36	0.44	0.49
	20° / 0.3m	0.62	**0.67**	0.47	0.56	0.56	0.61
Staircase Extended	10° / 0.1m	0.07	0.01	0.01	0.04	**0.08**	**0.08**
	15° / 0.2m	0.30	0.06	0.09	0.18	0.35	**0.40**
	20° / 0.3m	0.48	0.13	0.14	0.36	0.55	**0.60**
Seminar Room	10° / 0.1m	0.34	0.24	0.30	0.21	0.34	**0.42**
	15° / 0.2m	0.74	0.63	0.65	0.65	0.76	**0.77**
	20° / 0.3m	0.84	0.79	0.76	0.82	0.88	**0.90**
Average	10° / 0.1m	0.20	0.19	0.09	0.13	0.21	**0.23**
	15° / 0.2m	0.51	0.45	0.27	0.43	0.53	**0.56**
	20° / 0.3m	0.63	0.55	0.35	0.57	**0.64**	0.63

less degrees of freedom suffers from discontinuities in the rotation space that might harm the performance of deep learning models. Instead, [247], propose a continuous 6D or 5D representation. Since for rotation matrices the last column can be computed by the cross-product of the remaining two columns, it is unnecessary to predict all nine parameters of the matrix. Therefore to obtain the desired 6D representation, the last column can simply be dropped. To map back from a 6D prediction to a valid rotation matrix, the authors of [247] propose a Gram-Schmidt like orthogonalization procedure, which in case of rotation matrices results in computing the cross product of the first two normalized columns. Further the authors show that the proposed representation outperforms in comparison to euler angles, quaternions and axis angle representation when learned with a neural network.

To evaluate our method in this context, we map all poses to the proposed 6D representation and model them using a Gaussian mixture model, similar to a MDN, but treating rotation and translation separately. Therefore for each camera pose, in total we have 18 parameters to regress, plus mixture coefficients. Table 9.14 shows our results. 'Geo + ℓ_1' refers to a direct regression model using the geodesic loss proposed in [247] and an ℓ_1 loss on the translation. When using the proposed 6D representation, we found either improvements or similar performance to their quaternion counterparts. However, overall our 9D-Ours-RWTA and 9D-Ours-MBDN+RWTA remain the most promising models. Thus, the specific training scheme of our model can, regardless of the underlying distribution used, stabilize the training procedure and avoid mode collapse and therefore improve upon the baseline methods.

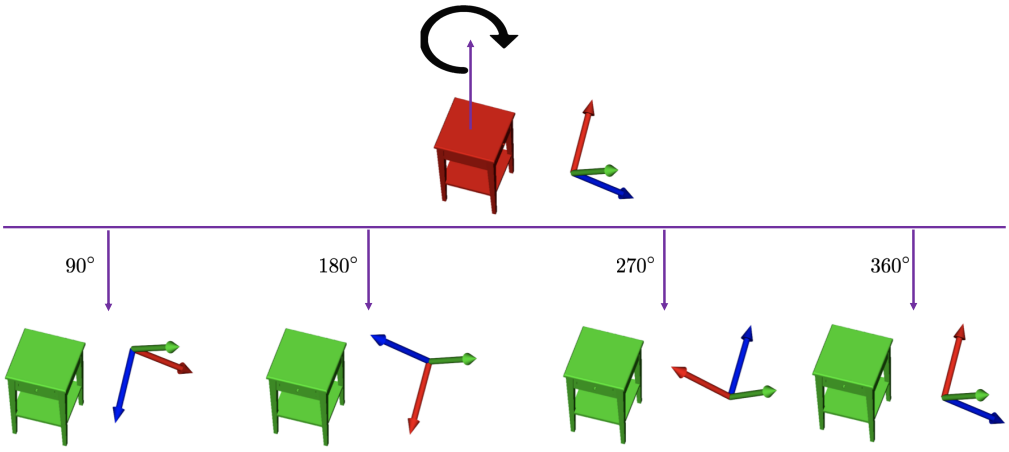


Fig. 9.17. Symmetric objects under rotations around their axis of symmetry introduce ambiguous views, that can result in erroneous predictions of deep learning models.

Remarks on Camera Localization

We have presented a novel method dealing with problems of direct camera pose regression methods in highly ambiguous environments where a unique solution to the 6DoF localization might be nonexistent. Instead, we predict camera pose hypotheses as well as associated uncertainties that finally form a mixture model. We use the Bingham distribution to model rotations and multivariate Gaussians to obtain the position of a camera. In contrast to other methods like MC-Dropout [106] or mixture density networks our training scheme is able to avoid mode collapse. Thus, we can obtain better mode predictions and improve upon the performance of camera pose regression methods in ambiguous environments while retaining the performance in non-ambiguous ones. We have proposed a general approach that we believe has the potential to foster research in various applications. Therefore, we now apply our method on an additional application, namely point cloud pose estimation.

9.8 Point Cloud Pose Estimation

With the advancements in autonomous driving and 3D capture, the need for efficient and accurate processing of 3D data has become a critical issue. At the backbone of numerous 3D systems lies point cloud processing, an essential tool for understanding the shapes surrounding us in 3D environments. In many of the cases, as already observed in 2D image views, objects exhibit symmetries and are observed under varying orientations that can result in ambiguities. In these situations the machinery of perception has to be made robust to the variations in the input that do not alter the object geometry. We illustrate this problem in Figure 9.17.

For this aim, we now evaluate our proposed Bingham Model in the context of object pose estimation from 3D point clouds and show that our model is well suitable to capture the multi-modality of symmetric objects in term of orientation in 3D as well.

Experimental Setup Therefore, we use objects of the ModelNet10 [228] dataset, which consists of ten object classes with example objects within each class. We randomly sample

Tab. 9.15. Point cloud pose estimation results on ModelNet10. We report the chamfer distance, where we scale the values by 10^2 for better presentation.

	ℓ_1	MC-Dropout	Uni	MBDN	Ours-MBDN+RWTA				Ours-RWTA
Num. Hypotheses	1	50	1	50	5	10	25	50	50
Average	3.605	3.509	1.477	1.237	0.927	**0.838**	0.926	0.973	1.201

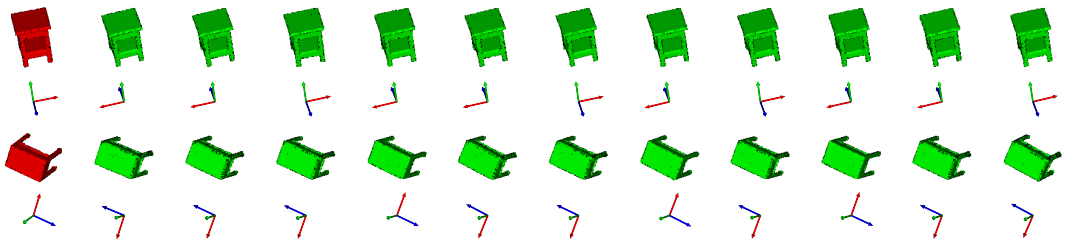


Fig. 9.18. Multiple hypotheses predictions of our network. The first column shows the object viewed from ground truth pose. The remaining columns show predicted poses (represented as a local reference frame) and the corresponding rotated object.

orientations and apply said orientations to the 3D point cloud to generate our training set. As our backbone network, we use PointNet [172] network architecture that has been widely used in recent literature to process point clouds with deep learning. The overall pipeline to predict uni-/multimodal Bingham distributions then remains the same. For this evaluation to construct $\mathbf{V}$ we remain with the method proposed by Birdal et al. [14].

To evaluate our method, we use the *Chamfer distance*. The chamfer distance measures the average distance over all points to the closest point in the target point cloud and is therefore well suited to capture the similarity between orientations of our point cloud samples. For instance, the angular error might not be as informative as large errors can easily occur in the presence of ambiguous views.

A summary of our results on the ModelNet10 dataset can be found in Table 9.15, where we report the results of our unimodal model (Uni), mixture model (MBDN) and MHP-versions, Ours-RWTA and Ours-MBDN+RWTA, in comparison to a simple regression model using an ℓ_1 loss and a MC-Dropout version of such a model. The results indicate that our models overall show better performance in comparison to the baselines, ℓ_1 and M-Dropout. Further multiple hypotheses prediction models, outperform a unimodal model, as ambiguous views can now be handled appropriately by the method.

Figure 9.18 shows qualitative results of our multiple hypotheses predictions, showing that our model is capable of predicting diverse, yet reasonable, orientations in comparison to the ground truth.

Based on our results, we can conclude that our method provides a general framework that is easily applicable to orientation regression methods on 2D as well as 3D input information. In this context, we leave further experiments such as evaluations on six DoF pose estimation on point cloud data as future work.

9.9 Conclusion

We have shown an extensive evaluation of our proposed Bingham Model as well as multiple hypothesis prediction and uncertainty estimation in the context of visual camera localization as well as point cloud pose estimation [56]. In highly ambiguous environments our method is well able to capture the multi-modal nature and uncertainty present in such scenarios that can confuse supervised deep learning models. Further, our method provides a general framework for pose regression and can easily be transferred to similar applications such as object pose estimation as well as to various input modalities ranging from 2D data to 3D point clouds.

Part V

Discussion and Conclusion

10 Discussion and Conclusion

10.1 Summary

In this dissertation, we have presented several frameworks addressing issues arising in current image-based localization methods by relying on deeply learned regression approaches.

First, we have introduced a multi-task learning framework that is trained to predict object relevant features. In addition, we have shown that providing information about the object's pose, in the form of a regression loss, enables the model to learn features that can more efficiently be distinguished and used for object recognition as well as pose retrieval. Furthermore, we show that our method is able to adapt well to related tasks such as place recognition in large-scale outdoor environments. Though, it has to be kept in mind that such methods always depend on the constructed database for retrieval and as such are limited by the database size in both accuracy and matching efficiency. The combination of such methods, however, for instance with regression approaches has shown promising results.

To alleviate this issue, next we explore pixel-wise scene coordinate regression assuming a reconstruction or depth information is available to obtain ground truth scene coordinates. As long as such a reconstruction of the scene, that covers its main features, is available, we can train scene specific models and theoretically are able to predict scene coordinates without restriction. Although due to inaccuracies of the model and noise in the data, outliers remain unavoidable. Therefore, for each regressed coordinate we additionally regress a corresponding confidence value in a second stage. These confidences are trained to be correlated to the error of the regressed coordinate such that outliers with high error can easily be removed by analyzing its confidence value. The confidences then provide a measure that can be used to rank camera pose hypothesis computed from the correspondences and used in a subsequent pose refinement.

Finally, we have moved towards methods relying solely on RGB information without requiring a reconstruction or 3D model of the scene and explore direct regression of the camera pose.

First, aiming at bridging the gap between the accuracy of direct pose regression methods and structure-based models, we have proposed a pose refinement strategy that is inspired by the framework of generative adversarial networks. Therefore providing an approach dependent only on RGB information with low computational cost.

Second, we analyze the performance of direct regression methods in the context of highly ambiguous environments. We proposed a method predicting full distributions on the 6D

pose space that is able to provide reliable uncertainty estimates as well as handle multiple possible solutions by incorporating a multiple hypotheses training strategy. While maintaining the benefits of regression approaches in non-ambiguous environments our method is able to handle ambiguous environments as well and in comparison to the state of the art introduces a negligible training and inference overhead.

10.2 Limitations and Future Work

Regression methods as demonstrated in this thesis offer great capabilities and potential as their range of output is theoretically unlimited. However, in practice such methods certainly show significant challenges and drawbacks. In context of deep learning, the training stage of such models strongly influences their overall performance and comes with certain problems as well. The lack of available training data has posed an ongoing challenge for almost all current deep learning solutions and a multitude of applications in computer vision. In the context of camera localization, obtaining a large amount of data with accurate ground truth camera poses is currently not feasible and models, as a result, often lack generalization capabilities. Few works have presented methods that synthesize additional views, thus, enlarging the dataset [171]. On the other hand, domain adaption techniques often deployed to handle such problems, for instance to train models on synthetic data rendered from a CAD model, are not straightforward to apply for the task at hand. Accurate and large scale models are often not available and rendering realistic image settings remains computationally demanding.

Similar, unsupervised methods, often deployed to handle the absence of large amounts of labels, have not yet been investigated for the task of camera localization and are not easily adapted to handle such a regression task either. Few works have addressed the use of additional information such as GPS or visual odometry to fine-tune a deep learning model in an unsupervised fashion [26], however, have not yet shown to solve the generalization problem or provide fully unsupervised or self-supervised training schemes.

With respect to SLAM applications, scenes are currently assumed to be static. However, this is rarely the case in real-world scenarios. Dynamic environments readily occur, but as of now current re-localization systems are not designed to handle even smaller changes in the scene. As a result, once changed the map has to be recreated to ensure adaptability to the changed environment. To enable fully autonomous systems, such methods will have to be robust to movements in the scene, such as objects changing their position and orientation, as well as to illumination changes and changing lighting conditions. In turn changes in the scene can introduce ambiguous views that, as shown in this thesis, can easily confuse deep learning models. A natural next step can be to extend our framework and analyze its ability to handle such cases. Uncertainty estimation plays a crucial role in this context and can be applied to detect changes in the environment as a first step to handle such scenarios. As the overall scene is usually not affected by major changes and most changes occur due to small scale objects or deformable content, uncertain regions should not influence a model's prediction to provide accurate and robust localization, a concept already applied in autonomous driving scenarios [94].

10.3 Conclusion

We believe we have demonstrated the potential of image-based localization methods in the context of camera localization by utilizing deep learning models. As current developments indicate visual sensors will play a crucial role in the development of fully autonomous systems. Image retrieval methods are always restricted to the database at hand, making regression approaches a promising solution to solve a variety of large scale tasks. However, drawbacks of such methods, that arise in deep learning frameworks have to be addressed with care such that these methods can develop and show their potential. We believe with the work presented in this thesis we have taken a step towards optimization of deep learning based regression methods and shown promising results for the example of camera pose estimation. Our methods, once trained, can be deployed fully automatic and can thus pave the way for more accurate as well as automatic computer vision systems in a variety of applications ranging from augmented reality to robotic navigation and autonomous driving.

Part VI

Appendix

List of Publications

A

1. **Mai Bui**, Tolga Birdal, Haowen Deng, Shadi Albarqouni, Leonidas Guibas, Slobodan Ilic, Nassir Navab, *'6D Camera Relocalization in Ambiguous Scenes via Continuous Multimodal Inference'*, Proceedings of the Europoean Conference on Computer Vision, 23-28 August, 2020, Glasgow

2. **Mai Bui***, Christoph Baur*, Nassir Navab, Slobodan Ilic**, Shadi Albarqouni**, *'Adversarial Networks for Camera Pose Regression and Refinement'*, Proceedings of the International Conference on Computer Vision Workshops (ICCVW), 27 October - 2 November, 2019, Seoul (* The first two authors contribute equally to this paper. **The last two authors share senior authorship.)

3. **Mai Bui**, Shadi Albarqouni, Slobodan Ilic, Nassir Navab, *'Scene Coordinate and Correspondence Learning for Image-Based Localization'*, Proceedings of the British Machine Vision Conference (BMVC), 3-6 September, 2018, Newcastle (Oral presentation)

4. **Mai Bui**, Sergey Zakharov, Shadi Albarqouni, Slobodan Ilic, Nassir Navab, *'When Regression meets Manifold Learning for Object Recognition and Pose Estimation'*, Proceedings of IEEE International Conference on Robotics and Automation (ICRA), 21-25 May, 2018, Brisbane, Australia

Additional Publications

B

1. **Mai Bui***, Shadi Albarqouni*, Michael Schrapp, Nassir Navab, Slobodan Ilic, *'X-ray PoseNet: 6 DoF Pose Estimation for Mobile X-ray Devices'*, Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV), Mar 24, 2017 - Mar 31, 2017, Santa Rosa, USA (* The first two authors contribute equally to this paper.) [32]

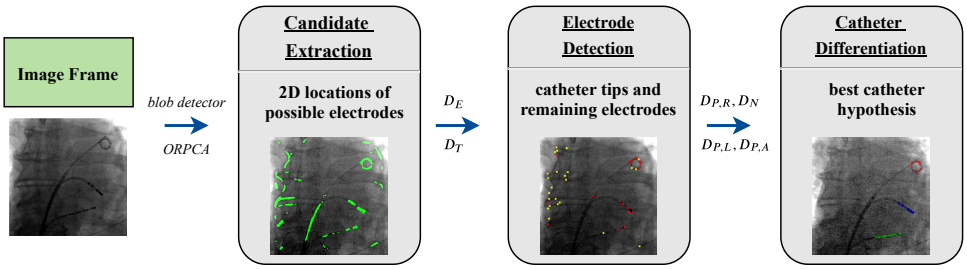
2. **Mai Bui**, Felix Bourier, Christoph Baur, Fausto Milletari, Nassir Navab, Stefanie Demirci, *'Robust Navigation Support in Lowest Dose Image Setting'*, International Journal of Computer Assisted Radiology and Surgery (IJCARS), 2018 [35]

C

Abstracts of Additional Publications

Robust Navigation Support in Lowest Dose Image Settings

Mai Bui, Felix Bourier, Christoph Baur, Fausto Milletari, Nassir Navab, Stefanie Demirci



Purpose Clinical cardiac electro-physiology (EP) is concerned with diagnosis and treatment of cardiac arrhythmia describing abnormality or perturbation in the normal activation sequence of the myocardium. With the recent introduction of lowest dose X-ray imaging protocol for EP procedures, interventional image enhancement has gained crucial importance for the well-being of patients as well as medical staff.

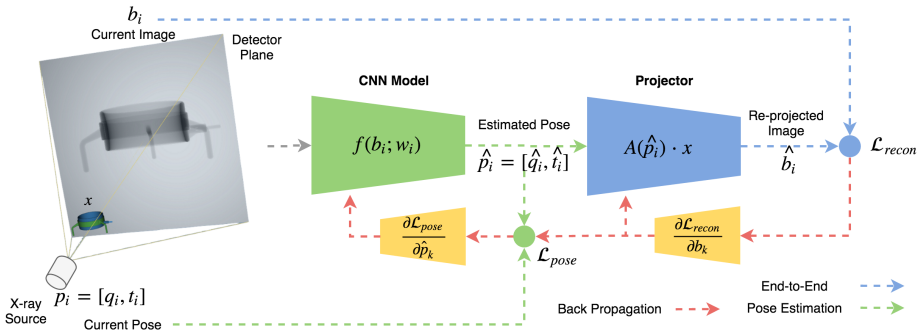
Methods In this paper, we introduce a novel method to detect and track different EP catheter electrodes in lowest dose fluoroscopic sequences based on l_1-sparse coding and online robust PCA (ORPCA). Besides being able to work on real lowest dose sequences, the underlying methodology achieves simultaneous detection and tracking of three main EP catheters used during ablation procedures.

Results We have validated our algorithm on 16 lowest dose fluoroscopic sequences acquired during real cardiac ablation procedures. In addition to expert labels for 2 sequences, we have employed a crowdsourcing strategy to obtain ground truth labels for the remaining 14 sequences. In order to validate the effect of different training data, we have employed a leave-one-out cross-validation scheme yielding an average detection rate of 86.9%.

Conclusion Besides these promising quantitative results, our medical partners also expressed their high satisfaction. Being based on l_1-sparse coding and online robust PCA (ORPCA), our method advances previous approaches by being able to detect and track electrodes attached to multiple different catheters. [35]

X-ray PoseNet: 6 DoF Pose Estimation for Mobile X-ray Devices

Mai Bui, Shadi Albarqouni, Michael Schrapp, Nassir Navab, Slobodan Ilic



Precise reconstruction of 3D volumes from X-ray projections requires precisely pre-calibrated systems where accurate knowledge of the systems geometric parameters is known ahead. However, when dealing with mobile X-ray devices such calibration parameters are unknown. Joint estimation of the systems calibration parameters and 3d reconstruction is a heavily unconstrained problem, especially when the projections are arbitrary. In industrial applications, that we target here, nominal CAD models of the object to be reconstructed are usually available. We rely on this prior information and employ Deep Learning to learn the mapping between simulated X-ray projections and its pose. Moreover, we introduce the reconstruction loss in addition to the pose loss to further improve the reconstruction quality. Finally, we demonstrate the generalization capabilities of our method in case where poses can be learned on instances of the objects belonging to the same class, allowing pose estimation of unseen objects from the same category, thus eliminating the need for the actual CAD model. We performed exhaustive evaluation demonstrating the quality of our results on both synthetic and real data. [32]

Bibliography

[1] M. Abadi, P. Barham, J. Chen, et al. “Tensorflow: A system for large-scale machine learning”. In: *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*. 2016, pp. 265–283 (cit. on pp. 34, 60).

[2] M. Angelina Uy and G. Hee Lee. “Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4470–4479 (cit. on p. 25).

[3] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. “NetVLAD: CNN architecture for weakly supervised place recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 5297–5307 (cit. on p. 25).

[4] T. Bailey and H. Durrant-Whyte. “Simultaneous localisation and mapping (SLAM): Part i the essential algorithms”. In: *IEEE Robotics and Automation Magazine* 13.2 (2006), pp. 99–110 (cit. on p. 7).

[5] V. Balntas, A. Doumanoglou, C. Sahin, J. Sock, R. Kouskouridas, and T.-K. Kim. “Pose guided RGBD feature learning for 3D object pose estimation”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 3856–3864 (cit. on p. 26).

[6] V. Balntas, S. Li, and V. Prisacariu. “Relocnet: Continuous metric learning relocalisation using neural nets”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 751–767 (cit. on pp. 71, 72, 111).

[7] T. D. Barfoot and P. T. Furgale. “Associating uncertainty with three-dimensional poses for use in estimation problems”. In: *IEEE Transactions on Robotics* 30.3 (2014), pp. 679–693 (cit. on p. 94).

[8] H. Bay, T. Tuytelaars, and L. Van Gool. “Surf: Speeded up robust features”. In: *European conference on computer vision*. Springer. 2006, pp. 404–417 (cit. on p. 23).

[9] V. Belagiannis, C. Rupprecht, G. Carneiro, and N. Navab. “Robust optimization for deep regression”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2830–2838 (cit. on p. 57).

[10] P. J. Besl and N. D. McKay. “Method for registration of 3-D shapes”. In: *Sensor fusion IV: control paradigms and data structures*. Vol. 1611. International Society for Optics and Photonics. 1992, pp. 586–606 (cit. on p. 74).

[11] J. Bian, Z. Li, N. Wang, et al. “Unsupervised scale-consistent depth and ego-motion learning from monocular video”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 35–45 (cit. on p. 10).

[12] A. Bicchi and V. Kumar. “Robotic grasping and contact: A review”. In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*. Vol. 1. IEEE. 2000, pp. 348–353 (cit. on p. 6).

[13] C. Bingham. “An antipodally symmetric distribution on the sphere”. In: *The Annals of Statistics* (1974), pp. 1201–1225 (cit. on p. 92).

[14] T. Birdal, U. Simsekli, M. O. Eken, and S. Ilic. “Bayesian Pose Graph Optimization via Bingham Distributions and Tempered Geodesic MCMC”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 308–319 (cit. on pp. 79, 93, 116).

[15] C. M. Bishop. “Mixture density networks”. In: (1994) (cit. on pp. 91, 100, 102).

[16] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison. “CodeSLAM—learning a compact, optimisable representation for dense visual SLAM”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 2560–2568 (cit. on p. 8).

[17] K. Bousmalis, A. Irpan, P. Wohlhart, et al. “Using simulation and domain adaptation to improve efficiency of deep robotic grasping”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 4243–4250 (cit. on p. 27).

[18] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. “Unsupervised pixel-level domain adaptation with generative adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 3722–3731 (cit. on p. 27).

[19] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. “Domain separation networks”. In: *Advances in neural information processing systems*. 2016, pp. 343–351 (cit. on p. 27).

[20] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas. “Probabilistic data association for semantic slam”. In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2017, pp. 1722–1729 (cit. on p. 9).

[21] E. Brachmann, A. Krull, S. Nowozin, et al. “DSAC-differentiable RANSAC for camera localization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6684–6692 (cit. on pp. 9, 52, 54, 58–60, 64, 65, 76, 85).

[22] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold, et al. “Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 3364–3372 (cit. on p. 91).

[23] E. Brachmann and C. Rother. “Expert sample consensus applied to camera re-localization”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 7525–7534 (cit. on p. 54).

[24] E. Brachmann and C. Rother. “Learning less is more-6d camera localization via 3d surface regression”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4654–4662 (cit. on pp. 9, 52, 54, 65, 76, 86).

[25] E. Brachmann and C. Rother. “Neural-guided RANSAC: Learning where to sample model hypotheses”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 4322–4331 (cit. on pp. 54, 65).

[26] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz. “Geometry-aware learning of maps for camera localization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 2616–2625 (cit. on pp. 70, 71, 77, 80, 81, 86, 102–104, 111, 113, 122).

[27] S. Brahmbhatt, C. Ham, C. C. Kemp, and J. Hays. “ContactDB: Analyzing and predicting grasp contact via thermal imaging”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8709–8719 (cit. on p. 6).

[28] L. Breiman. “Random forests”. In: *Machine learning* 45.1 (2001), pp. 5–32 (cit. on p. 91).

[29] I. Budvytis, P. Sauer, and R. Cipolla. “Semantic localisation via globally unique instance segmentation”. In: (2018) (cit. on p. 25).

[30] I. Budvytis, M. Teichmann, T. Vojir, and R. Cipolla. “Large scale joint semantic re-localisation and scene understanding via globally unique instance coordinate regression”. In: (2019) (cit. on p. 54).

[31] M. Bui, S. Albarqouni, S. Ilic, and N. Navab. “Scene Coordinate and Correspondence Learning for Image-Based Localization”. In: *British Machine Vision Conference (BMVC)*. 2018 (cit. on p. 55).

[32] M. Bui, S. Albarqouni, M. Schrapp, N. Navab, and S. Ilic. “X-Ray PoseNet: 6 DoF Pose Estimation for Mobile X-Ray Devices”. In: *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*. IEEE. 2017, pp. 1036–1044 (cit. on pp. 37, 129, 132).

[33] M. Bui, C. Baur, N. Navab, S. Ilic, and S. Albarqouni. “Adversarial Networks for Camera Pose Regression and Refinement”. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2019, pp. 0–0 (cit. on p. 76).

[34] M. Bui, T. Birdal, H. Deng, et al. “6D Camera Relocalization in Ambiguous Scenes via Continuous Multimodal Inference”. In: *European Conference on Computer Vision*. 2020 (cit. on p. 90).

[35] M. Bui, F. Bourier, C. Baur, F. Milletari, N. Navab, and S. Demirci. “Robust navigation support in lowest dose image setting”. In: *International journal of computer assisted radiology and surgery* 14.2 (2019), pp. 291–300 (cit. on pp. 129, 131).

[36] M. Bui, S. Zakharov, S. Albarqouni, S. Ilic, and N. Navab. “When regression meets manifold learning for object recognition and pose estimation”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 1–7 (cit. on p. 29).

[37] S. Byrne and M. Girolami. “Geodesic Monte Carlo on embedded manifolds”. In: *Scandinavian Journal of Statistics* 40.4 (2013), pp. 825–845 (cit. on p. 79).

[38] M. Cai, C. Shen, and I. D. Reid. “A Hybrid Probabilistic Model for Camera Relocalization.” In: *BMVC*. Vol. 1. 2. 2018, p. 8 (cit. on pp. 71, 91).

[39] J. A. Castellanos and J. D. Tardos. *Mobile robot localization and map building: A multisensor fusion approach*. Springer Science & Business Media, 2012 (cit. on p. 7).

[40] T. Cavallari, L. Bertinetto, J. Mukhoti, P. Torr, and S. Golodetz. “Let’s Take This Online: Adapting Scene Coordinate Regression Network Predictions for Online RGB-D Camera Relocalisation”. In: *2019 International Conference on 3D Vision (3DV)*. IEEE. 2019, pp. 564–573 (cit. on pp. 52, 54).

[41] T. Cavallari, S. Golodetz, N. Lord, et al. “Real-time RGB-D camera pose estimation in novel scenes using a relocalisation cascade”. In: *IEEE transactions on pattern analysis and machine intelligence* (2019) (cit. on pp. 18, 52, 53).

[42] T. Cavallari, S. Golodetz, N. A. Lord, J. Valentin, L. Di Stefano, and P. H. Torr. “On-the-fly adaptation of regression forests for online camera relocalisation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 4457–4466 (cit. on pp. 9, 52, 53).

[43] H. Chen, A. S. Lee, M. Swift, and J. C. Tang. “3D collaboration method over HoloLens™ and Skype™ end points”. In: *Proceedings of the 3rd International Workshop on Immersive Media Experiences*. 2015, pp. 27–30 (cit. on p. 6).

[44] J. X. Chen. “The evolution of computing: AlphaGo”. In: *Computing in Science & Engineering* 18.4 (2016), pp. 4–7 (cit. on p. 3).

[45] Y. Chen and G. G. Medioni. “Object modeling by registration of multiple range images.” In: *Image Vision Comput.* 10.3 (1992), pp. 145–155 (cit. on p. 74).

[46] Y. Chen, C. Shen, X.-S. Wei, L. Liu, and J. Yang. “Adversarial posenet: A structure-aware convolutional network for human pose estimation”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 1212–1221 (cit. on pp. 75, 76).

[47] Y. Chen, C. Schmid, and C. Sminchisescu. “Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 7063–7072 (cit. on p. 10).

[48] Z. Chen, A. Jacobson, N. Sünderhauf, et al. “Deep learning features at scale for visual place recognition”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 3223–3230 (cit. on p. 25).

[49] Z. Chen, F. Maffra, I. Sa, and M. Chli. “Only look once, mining distinctive landmarks from convnet for visual place recognition”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 9–16 (cit. on p. 25).

[50] R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen. “Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6856–6864 (cit. on pp. 70, 71, 91, 102–104).

[51] E. Corona, K. Kundu, and S. Fidler. “Pose Estimation for Objects with Rotational Symmetry”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 7215–7222 (cit. on p. 90).

[52] G. Csurka. “A Comprehensive Survey on Domain Adaptation for Visual Applications”. In: *Domain Adaptation in Computer Vision Applications*. Ed. by G. Csurka. Cham: Springer International Publishing, 2017, pp. 1–35 (cit. on p. 27).

[53] H. Cui, V. Radosavljevic, F.-C. Chou, et al. “Multimodal trajectory predictions for autonomous driving using deep convolutional networks”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 2090–2096 (cit. on p. 91).

[54] N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection”. In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR’05)*. Vol. 1. IEEE. 2005, pp. 886–893 (cit. on p. 23).

[55] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. “MonoSLAM: Real-time single camera SLAM”. In: *IEEE transactions on pattern analysis and machine intelligence* 29.6 (2007), pp. 1052–1067 (cit. on p. 7).

[56] H. Deng, M. Bui, N. Navab, L. Guibas, S. Ilic, and T. Birdal. “Deep Bingham Networks: Dealing with Uncertainty and Ambiguity in Pose Estimation”. In: *arXiv preprint arXiv:2012.11002* (2020) (cit. on p. 117).

[57] D. DeTone, T. Malisiewicz, and A. Rabinovich. “Superpoint: Self-supervised interest point detection and description”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018, pp. 224–236 (cit. on p. 65).

[58] M. Ding, Z. Wang, J. Sun, J. Shi, and P. Luo. “CamNet: Coarse-to-Fine Retrieval for Camera Re-Localization”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 2871–2880 (cit. on p. 72).

[59] M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba. “A solution to the simultaneous localization and map building (SLAM) problem”. In: *IEEE Transactions on robotics and automation* 17.3 (2001), pp. 229–241 (cit. on p. 7).

[60] T.-T. Do, T. Pham, M. Cai, and I. Reid. “Real-time monocular object instance 6d pose estimation”. In: *British Machine Vision Conference (BMVC)*. Vol. 1. 2. 2018, p. 6 (cit. on p. 77).

[61] M. Dusmanu, I. Rocco, T. Pajdla, et al. “D2-Net: A Trainable CNN for Joint Description and Detection of Local Features”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8092–8101 (cit. on p. 52).

[62] E. Eade. “Lie groups for 2d and 3d transformations”. In: *URL http://ethaneade. com/lie. pdf, revised Dec* (2013) (cit. on pp. 14, 16).

[63] J. Engel, T. Schöps, and D. Cremers. “LSD-SLAM: Large-scale direct monocular SLAM”. In: *European conference on computer vision*. Springer. 2014, pp. 834–849 (cit. on p. 8).

[64] M. Everingham and J. Winn. “The PASCAL visual object classes challenge 2012 (VOC2012) development kit”. In: *Pattern Analysis, Statistical Modelling and Computational Learning, Tech. Rep* (2011) (cit. on p. 27).

[65] L. Falorsi, P. de Haan, T. R. Davidson, and P. Forré. “Reparameterizing Distributions on Lie Groups”. In: *arXiv preprint arXiv:1903.02958* (2019) (cit. on p. 94).

[66] M. A. Fischler and R. C. Bolles. “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography”. In: *Communications of the ACM* 24.6 (1981), pp. 381–395 (cit. on pp. 17, 91).

[67] R. Frampton and A. Calway. “Place recognition from disparate views.” In: *BMVC*. 2013 (cit. on p. 9).

[68] Y. Gal and Z. Ghahramani. “Dropout as a bayesian approximation: Representing model uncertainty in deep learning”. In: *international conference on machine learning*. 2016, pp. 1050–1059 (cit. on p. 91).

[69] D. Gálvez-López and J. D. Tardos. “Bags of binary words for fast place recognition in image sequences”. In: *IEEE Transactions on Robotics* 28.5 (2012), pp. 1188–1197 (cit. on p. 9).

[70] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. “Vision meets Robotics: The KITTI Dataset”. In: *International Journal of Robotics Research (IJRR)* (2013) (cit. on p. 9).

[71] M. Geppert, P. Liu, Z. Cui, M. Pollefeys, and T. Sattler. “Efficient 2d-3d matching for multi-camera visual localization”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 5972–5978 (cit. on p. 53).

[72] G. Ghazaei, I. Laina, C. Rupprecht, F. Tombari, N. Navab, and K. Nazarpour. “Dealing with ambiguity in robotic grasping via multiple predictions”. In: *Asian Conference on Computer Vision*. Springer. 2018, pp. 38–55 (cit. on p. 6).

[73] I. Gilitschenski, R. Sahoo, W. Schwarting, A. Amini, S. Karaman, and D. Rus. “Deep Orientation Uncertainty Learning based on a Bingham Loss”. In: *International Conference on Learning Representations*. 2020 (cit. on pp. 94, 102).

[74] R. Girshick. “Fast r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448 (cit. on p. 6).

[75] B. Glocker, J. Shotton, A. Criminisi, and S. Izadi. “Real-time RGB-D camera relocalization via randomized ferns for keyframe encoding”. In: *IEEE transactions on visualization and computer graphics* 21.5 (2014), pp. 571–583 (cit. on p. 9).

[76] J. Glover and L. P. Kaelbling. “Tracking the spin on a ping pong ball with the quaternion Bingham filter”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 2014, pp. 4133–4140 (cit. on p. 93).

[77] J. Glover, G. Bradski, and R. B. Rusu. “Monte carlo pose estimation with quaternion kernels and the bingham distribution”. In: *Robotics: science and systems*. Vol. 7. 2012, p. 97 (cit. on p. 93).

[78] G. Godin, M. Rioux, and R. Baribeau. “Three-dimensional registration using range and intensity information”. In: *Videometrics III*. Vol. 2350. International Society for Optics and Photonics. 1994, pp. 279–290 (cit. on p. 74).

[79] I. Goodfellow, J. Pouget-Abadie, M. Mirza, et al. “Generative adversarial nets”. In: *Advances in neural information processing systems*. 2014, pp. 2672–2680 (cit. on pp. 75, 76).

[80] F. S. Grassia. “Practical parameterization of rotations using the exponential map”. In: *Journal of graphics tools* 3.3 (1998), pp. 29–48 (cit. on p. 94).

[81] S. Griffith, G. Chahine, and C. Pradalier. “Symphony lake dataset”. In: *The International Journal of Robotics Research* 36.11 (2017), pp. 1151–1158 (cit. on p. 43).

[82] J. A. Grunert. “Das pothenotische problem in erweiterter gestalt nebst bber seine anwendungen in der geodasie”. In: *Grunerts Archiv fur Mathematik und Physik* (1841), pp. 238–248 (cit. on p. 17).

[83] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. “On calibration of modern neural networks”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 1321–1330 (cit. on p. 91).

[84] J. Guo, P. V. Borges, C. Park, and A. Gawel. “Local descriptor for robust place recognition using LiDAR intensity”. In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 1470–1477 (cit. on p. 25).

[85] A. Guzman-Rivera, D. Batra, and P. Kohli. “Multiple choice learning: Learning to produce multiple structured outputs”. In: *Advances in Neural Information Processing Systems*. 2012, pp. 1799–1807 (cit. on p. 96).

[86] A. Guzman-Rivera, P. Kohli, B. Glocker, et al. “Multi-output learning for camera relocalization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 1114–1121 (cit. on pp. 52, 53).

[87] K. He, X. Zhang, S. Ren, and J. Sun. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778 (cit. on pp. 85, 91, 95, 101).

[88] C. S. Herz. “Bessel Functions of Matrix Argument”. In: *Annals of Mathematics* 61.3 (1955), pp. 474–523 (cit. on p. 93).

[89] J. A. Hesch and S. I. Roumeliotis. “A direct least-squares (DLS) method for PnP”. In: *2011 International Conference on Computer Vision*. IEEE. 2011, pp. 383–390 (cit. on p. 17).

[90] S. Hinterstoisser, V. Lepetit, S. Ilic, et al. “Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes”. In: *Asian conference on computer vision*. Springer. 2012 (cit. on pp. 6, 31, 38).

[91] J. Ho and S. Ermon. “Generative adversarial imitation learning”. In: *Advances in neural information processing systems*. 2016, pp. 4565–4573 (cit. on p. 6).

[92] T. Hodaň, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, and X. Zabulis. “T-LESS: An RGB-D Dataset for 6D Pose Estimation of Texture-less Objects”. In: *IEEE Winter Conference on Applications of Computer Vision (WACV)* (2017) (cit. on p. 98).

[93] K. Hsiao and T. Lozano-Perez. “Imitation learning of whole-body grasps”. In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2006, pp. 5657–5662 (cit. on p. 6).

[94] Z. Huang, Y. Xu, J. Shi, X. Zhou, H. Bao, and G. Zhang. “Prior guided dropout for robust visual localization in dynamic environments”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 2791–2800 (cit. on pp. 71, 91, 122).

[95] D. Q. Huynh. “Metrics for 3D rotations: Comparison and analysis”. In: *Journal of Mathematical Imaging and Vision* 35.2 (2009), pp. 155–164 (cit. on p. 15).

[96] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. “Image-to-image translation with conditional adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134 (cit. on pp. 75, 80).

[97] M. Jaderberg, K. Simonyan, A. Zisserman, et al. “Spatial transformer networks”. In: *Advances in neural information processing systems*. 2015, pp. 2017–2025 (cit. on p. 59).

[98] S. James, P. Wohlhart, M. Kalakrishnan, et al. “Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 12627–12637 (cit. on p. 27).

[99] H. Jégou, M. Douze, C. Schmid, and P. Pérez. “Aggregating local descriptors into a compact image representation”. In: *CVPR 2010-23rd IEEE Conference on Computer Vision & Pattern Recognition*. IEEE Computer Society. 2010, pp. 3304–3311 (cit. on p. 25).

[100] W. Kabsch. “A solution for the best rotation to relate two sets of vectors”. In: *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography* 32.5 (1976), pp. 922–923 (cit. on p. 16).

[101] D. Kalashnikov, A. Irpan, P. Pastor, et al. “Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation”. In: *Conference on Robot Learning* (2018) (cit. on p. 6).

[102] A. Katiyar, K. Kalra, and C. Garg. “Marker based augmented reality”. In: *Advances in Computer Science and Information Technology (ACSIT)* 2.5 (2015), pp. 441–445 (cit. on p. 5).

[103] T. Ke and S. I. Roumeliotis. “An efficient algebraic solution to the perspective-three-point problem”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 7225–7233 (cit. on p. 17).

[104] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. “Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 1521–1529 (cit. on pp. 6, 73, 90).

[105] A. Kendall and R. Cipolla. “Geometric loss functions for camera pose regression with deep learning”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 5974–5983 (cit. on pp. 9, 64, 70, 77, 86, 102–104).

[106] A. Kendall and R. Cipolla. “Modelling uncertainty in deep learning for camera relocalization”. In: *2016 IEEE international conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 4762–4769 (cit. on pp. 71, 91, 102–104, 106, 115).

[107] A. Kendall, M. Grimes, and R. Cipolla. “Posenet: A convolutional network for real-time 6-dof camera relocalization”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2938–2946 (cit. on pp. 9, 30, 70, 75, 81, 86, 101, 103, 104, 106, 113).

[108] C. Kerl, J. Sturm, and D. Cremers. “Dense visual SLAM for RGB-D cameras”. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 2100–2106 (cit. on p. 8).

[109] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim. “Learning to discover cross-domain relations with generative adversarial networks”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 1857–1865 (cit. on p. 27).

[110] G. Klein and D. Murray. “Parallel tracking and mapping for small AR workspaces”. In: *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE. 2007, pp. 225–234 (cit. on pp. 5, 8).

[111] G. Klein and D. Murray. “Parallel tracking and mapping on a camera phone”. In: *2009 8th IEEE International Symposium on Mixed and Augmented Reality*. IEEE. 2009, pp. 83–86 (cit. on p. 5).

[112] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105 (cit. on pp. 23, 85).

[113] A. Kume and A. T. Wood. “Saddlepoint approximations for the Bingham and Fisher–Bingham normalising constants”. In: *Biometrika* 92.2 (2005), pp. 465–476 (cit. on p. 93).

[114] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. “g 2 o: A general framework for graph optimization”. In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 3607–3613 (cit. on p. 7).

[115] G. Kurz, I. Gilitschenski, S. Julier, and U. D. Hanebeck. “Recursive estimation of orientation based on the Bingham distribution”. In: *Information Fusion (FUSION), 2013 16th International Conference on*. IEEE. 2013, pp. 1487–1494 (cit. on p. 93).

[116] G. Kurz, I. Gilitschenski, F. Pfaff, et al. “Directional Statistics and Filtering Using libDirectional”. In: *arXiv preprint arXiv:1712.09718* (2017) (cit. on p. 93).

[117] M. Labbé and F. Michaud. “RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation”. In: *Journal of Field Robotics* 36.2 (2019), pp. 416–446 (cit. on p. 102).

[118] M. Larsson, E. Stenborg, C. Toft, L. Hammarstrand, T. Sattler, and F. Kahl. “Fine-grained segmentation networks: Self-supervised segmentation for improved long-term visual localization”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 31–41 (cit. on p. 10).

[119] S. M. LaValle, A. Yershova, M. Katsev, and M. Antonov. “Head tracking for the Oculus Rift”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 187–194 (cit. on p. 5).

[120] T.-W. Lee and M. S. Lewicki. “Unsupervised image classification, segmentation, and enhancement using ICA mixture models”. In: *IEEE Transactions on Image Processing* 11.3 (2002), pp. 270–279 (cit. on p. 91).

[121] I. Lenz, H. Lee, and A. Saxena. “Deep learning for detecting robotic grasps”. In: *The International Journal of Robotics Research* 34.4-5 (2015), pp. 705–724 (cit. on p. 6).

[122] V. Lepetit, F. Moreno-Noguer, and P. Fua. “Epnp: An accurate o (n) solution to the pnp problem”. In: *International journal of computer vision* 81.2 (2009), p. 155 (cit. on pp. 17, 56, 60).

[123] C. Ley and T. Verdebout. “Directional Statistics in Machine Learning: A Brief Review Suvrit Sra”. In: *Applied Directional Statistics*. Chapman and Hall/CRC, 2018, pp. 275–292 (cit. on p. 95).

[124] J. Li, D. Meger, and G. Dudek. “Semantic Mapping for View-Invariant Relocalization”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 7108–7115 (cit. on p. 9).

[125] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox. “Deepim: Deep iterative matching for 6d pose estimation”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 683–698 (cit. on p. 76).

[126] C.-H. Lin, E. Yumer, O. Wang, E. Shechtman, and S. Lucey. “St-gan: Spatial transformer generative adversarial networks for image compositing”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 9455–9464 (cit. on pp. 75, 76).

[127] N. Linder and P. Maes. “LuminAR: portable robotic augmented reality interface design and prototype”. In: *Adjunct proceedings of the 23nd annual ACM symposium on User interface software and technology*. 2010, pp. 395–396 (cit. on p. 6).

[128] J. Linowes and K. Babilinski. *Augmented Reality for Developers: Build practical augmented reality applications with Unity, ARCore, ARKit, and Vuforia*. Packt Publishing Ltd, 2017 (cit. on p. 5).

[129] W. Liu, D. Anguelov, D. Erhan, et al. “Ssd: Single shot multibox detector”. In: *European conference on computer vision*. Springer. 2016, pp. 21–37 (cit. on p. 6).

[130] D. G. Lowe. “Distinctive image features from scale-invariant keypoints”. In: *International journal of computer vision* 60.2 (2004), pp. 91–110 (cit. on p. 23).

[131] Q.-T. Luong and O. D. Faugeras. “The fundamental matrix: Theory, algorithms, and stability analysis”. In: *International journal of computer vision* 17.1 (1996), pp. 43–75 (cit. on p. 18).

[132] L. v. d. Maaten and G. Hinton. “Visualizing data using t-SNE”. In: *Journal of Machine Learning Research* 9.Nov (2008), pp. 2579–2605 (cit. on pp. 35, 38).

[133] O. Makansi, E. Ilg, O. Cicek, and T. Brox. “Overcoming Limitations of Mixture Density Networks: A Sampling and Fitting Framework for Multimodal Future Prediction”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 7144–7153 (cit. on pp. 91, 96, 103, 110).

[134] D Mané et al. *TensorBoard: TensorFlow's visualization toolkit, 2015* (cit. on p. 38).

[135] F. Manhardt, D. M. Arroyo, C. Rupprecht, et al. "Explaining the Ambiguity of Object Detection and 6D Pose from Visual Data". In: IEEE/CVF. 2019 (cit. on p. 90).

[136] F. Manhardt, W. Kehl, N. Navab, and F. Tombari. "Deep model-based 6d pose refinement in rgb". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 800–815 (cit. on p. 76).

[137] E. Marder-Eppstein. "Project tango". In: *ACM SIGGRAPH 2016 Real-Time Live!* 2016, pp. 25–25 (cit. on p. 5).

[138] D. Marr. "Vision: A computational investigation into the human representation and processing of visual information". In: (1982) (cit. on p. 3).

[139] M. Maskrey and W. Wang. "Understanding ARKit". In: *Pro iPhone Development with Swift 4*. Springer, 2018, pp. 389–418 (cit. on p. 5).

[140] D. Massiceti, A. Krull, E. Brachmann, C. Rother, and P. H. Torr. "Random forests versus Neural Networks—What's best for camera localization?" In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 5118–5125 (cit. on p. 53).

[141] T. Masuda, K. Sakaue, and N. Yokoya. "Registration and integration of multiple range images for 3-D model construction". In: *Proceedings of 13th international conference on pattern recognition*. Vol. 1. IEEE. 1996, pp. 879–883 (cit. on p. 74).

[142] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger. "Fusion++: Volumetric object-level slam". In: *2018 international conference on 3D vision (3DV)*. IEEE. 2018, pp. 32–41 (cit. on p. 9).

[143] G. J. McLachlan and K. E. Basford. *Mixture models: Inference and applications to clustering*. Vol. 84. M. Dekker New York, 1988 (cit. on p. 100).

[144] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu. "Relative camera pose estimation using convolutional neural networks". In: *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer. 2017, pp. 675–687 (cit. on pp. 71, 72).

[145] L. Meng, J. Chen, F. Tung, J. J. Little, J. Valentin, and C. W. de Silva. "Backtracking regression forests for accurate camera relocalization". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 6886–6893 (cit. on p. 53).

[146] L. Meng, F. Tung, J. J. Little, J. Valentin, and C. W. de Silva. "Exploiting points and lines in regression forests for RGB-D camera relocalization". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 6827–6834 (cit. on p. 53).

[147] M. Mirza and S. Osindero. "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784* (2014) (cit. on p. 75).

[148] V. Mnih, K. Kavukcuoglu, D. Silver, et al. "Human-level control through deep reinforcement learning". In: *nature* 518.7540 (2015), pp. 529–533 (cit. on p. 3).

[149] V. Mnih, K. Kavukcuoglu, D. Silver, et al. "Playing atari with deep reinforcement learning". In: *arXiv preprint arXiv:1312.5602* (2013) (cit. on p. 3).

[150] A Morawiec and D. Field. "Rodrigues parameterization for orientation and misorientation distributions". In: *Philosophical Magazine A* 73.4 (1996), pp. 1113–1130 (cit. on p. 94).

[151] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. "ORB-SLAM: a versatile and accurate monocular SLAM system". In: *IEEE transactions on robotics* 31.5 (2015), pp. 1147–1163 (cit. on p. 7).

[152] R. Mur-Artal and J. D. Tardós. "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras". In: *IEEE Transactions on Robotics* 33.5 (2017), pp. 1255–1262 (cit. on p. 8).

[153] K. Murphy, A. Torralba, D. Eaton, and W. Freeman. “Object detection and localization using local and global features”. In: *Toward Category-Level Object Recognition*. Springer, 2006, pp. 382–400 (cit. on p. 91).

[154] R. M. Murray. *A mathematical introduction to robotic manipulation*. CRC press, 1994 (cit. on p. 94).

[155] U. Nadeem, M. A. Jalwana, M. Bennamoun, R. Togneri, and F. Sohel. “Direct Image to Point Cloud Descriptors Matching for 6-DOF Camera Localization in Dense 3D Point Clouds”. In: *International Conference on Neural Information Processing*. Springer. 2019, pp. 222–234 (cit. on p. 52).

[156] T. Naseer and W. Burgard. “Deep regression for monocular camera-based 6-dof global localization in outdoor environments”. In: *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE. 2017, pp. 1525–1530 (cit. on p. 60).

[157] N. Navab, A Bani-Kashemi, and M. Mitschke. “Merging visible and invisible: Two camera-augmented mobile C-arm (CAMC) applications”. In: *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR’99)*. IEEE. 1999, pp. 134–141 (cit. on p. 6).

[158] N. Navab, S.-M. Heining, and J. Traub. “Camera augmented mobile C-arm (CAMC): calibration, accuracy study, and clinical applications”. In: *IEEE transactions on medical imaging* 29.7 (2009), pp. 1412–1423 (cit. on p. 6).

[159] R. A. Newcombe, S. Izadi, O. Hilliges, et al. “KinectFusion: Real-time dense surface mapping and tracking”. In: *2011 10th IEEE International Symposium on Mixed and Augmented Reality*. IEEE. 2011, pp. 127–136 (cit. on pp. 73, 74).

[160] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. “DTAM: Dense tracking and mapping in real-time”. In: *2011 international conference on computer vision*. IEEE. 2011, pp. 2320–2327 (cit. on p. 8).

[161] D. C. Niehorster, L. Li, and M. Lappe. “The accuracy and precision of position and orientation tracking in the HTC vive virtual reality system for scientific research”. In: *i-Perception* 8.3 (2017), p. 2041669517708205 (cit. on p. 5).

[162] J. Pang and G. Cheung. “Graph Laplacian regularization for image denoising: analysis in the continuous domain”. In: *IEEE Transactions on Image Processing* 26.4 (2017), pp. 1770–1785 (cit. on p. 57).

[163] A. Paszke, S. Gross, S. Chintala, et al. “Automatic differentiation in pytorch”. In: (2017) (cit. on p. 80).

[164] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao. “Pvnet: Pixel-wise voting network for 6dof pose estimation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4561–4570 (cit. on p. 6).

[165] V. Peretroukhin, B. Wagstaff, M. Giamou, and J. Kelly. “Probabilistic Regression of Rotations using Quaternion Averaging and a Deep Multi-Headed Network”. In: *arXiv preprint arXiv:1904.03182* (2019) (cit. on p. 111).

[166] K. Perlin. “Noise hardware”. In: *Real-Time Shading SIGGRAPH Course Notes* (2001) (cit. on p. 34).

[167] P. Pessaux, M. Diana, L. Soler, T. Piardi, D. Mutter, and J. Marescaux. “Towards cybernetic surgery: robotic and augmented reality-assisted liver segmentectomy”. In: *Langenbeck’s archives of surgery* 400.3 (2015), pp. 381–385 (cit. on p. 6).

[168] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. “Object retrieval with large vocabularies and fast spatial matching”. In: *2007 IEEE conference on computer vision and pattern recognition*. IEEE. 2007, pp. 1–8 (cit. on p. 24).

[169] G. Pitteri, M. Ramamonjisoa, S. Ilic, and V. Lepetit. “On Object Symmetries and 6D Pose Estimation from Images”. In: IEEE. 2019 (cit. on p. 90).

[170] S. Prokudin, P. Gehler, and S. Nowozin. “Deep Directional Statistics: Pose Estimation with Uncertainty Quantification”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 534–551 (cit. on pp. 94, 95).

[171] P. Purkait, C. Zhao, and C. Zach. “Synthetic View Generation for Absolute Pose Regression and Image Synthesis.” In: *BMVC*. 2018, p. 69 (cit. on p. 122).

[172] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. “Pointnet: Deep learning on point sets for 3d classification and segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660 (cit. on pp. 25, 59, 116).

[173] M. Rad and V. Lepetit. “Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 3828–3836 (cit. on p. 90).

[174] A. Ranjan, V. Jampani, L. Balles, et al. “Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 12240–12249 (cit. on p. 10).

[175] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. “You only look once: Unified, real-time object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788 (cit. on pp. 6, 30).

[176] S. Ren, K. He, R. Girshick, and J. Sun. “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *Advances in neural information processing systems*. 2015, pp. 91–99 (cit. on p. 6).

[177] S. Riedel, Z.-C. Marton, and S. Kriegel. “Multi-view orientation estimation using Bingham mixture models”. In: *2016 IEEE international conference on automation, quality and testing, robotics (AQTR)*. IEEE. 2016, pp. 1–6 (cit. on p. 95).

[178] O. Ronneberger, P. Fischer, and T. Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241 (cit. on p. 59).

[179] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. “ORB: An efficient alternative to SIFT or SURF”. In: *2011 International conference on computer vision*. Ieee. 2011, pp. 2564–2571 (cit. on p. 7).

[180] Y. Rubner, C. Tomasi, and L. J. Guibas. “The earth mover’s distance as a metric for image retrieval”. In: *International journal of computer vision* 40.2 (2000), pp. 99–121 (cit. on p. 103).

[181] M. Runz, M. Buffier, and L. Agapito. “Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects”. In: *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE. 2018, pp. 10–20 (cit. on p. 9).

[182] C. Rupprecht, I. Laina, R. DiPietro, et al. “Learning in an uncertain world: Representing ambiguity through multiple hypotheses”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 3591–3600 (cit. on pp. 90, 95, 96, 110).

[183] S. Rusinkiewicz and M. Levoy. “Efficient variants of the ICP algorithm”. In: *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*. IEEE. 2001, pp. 145–152 (cit. on p. 74).

[184] O. Russakovsky, J. Deng, H. Su, et al. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252 (cit. on pp. 78, 87, 101).

[185] F. Sadeghi and S. Levine. “Cad2rl: Real single-image flight without a single real image”. In: *Robotics: Science and Systems* (2017) (cit. on p. 27).

[186] S. Saha, G. Varma, and C. Jawahar. “Improved visual relocalization by discovering anchor points”. In: *British Machine Vision Conference* (2018) (cit. on p. 70).

[187] H. Sahin and L. Guvenc. “Household robotics: autonomous devices for vacuuming and lawn mowing [applications of control]”. In: *IEEE Control Systems Magazine* 27.2 (2007), pp. 20–96 (cit. on p. 6).

[188] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. “Slam++: Simultaneous localisation and mapping at the level of objects”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2013, pp. 1352–1359 (cit. on p. 8).

[189] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk. “From coarse to fine: Robust hierarchical localization at large scale”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 12716–12725 (cit. on pp. 25, 26).

[190] T. Sattler, B. Leibe, and L. Kobbelt. “Efficient & effective prioritized matching for large-scale image-based localization”. In: *IEEE transactions on pattern analysis and machine intelligence* 39.9 (2016), pp. 1744–1756 (cit. on p. 52).

[191] T. Sattler, B. Leibe, and L. Kobbelt. “Fast image-based localization using direct 2d-to-3d matching”. In: *2011 International Conference on Computer Vision*. IEEE. 2011, pp. 667–674 (cit. on p. 52).

[192] T. Sattler, B. Leibe, and L. Kobbelt. “Improving image-based localization by active correspondence search”. In: *European conference on computer vision*. Springer. 2012, pp. 752–765 (cit. on p. 52).

[193] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe. “Understanding the Limitations of CNN-based Absolute Camera Pose Regression”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3302–3312 (cit. on pp. 71, 87, 103, 105).

[194] A. Saxena, J. Driemeyer, J. Kearns, and A. Y. Ng. “Robotic grasping of novel objects”. In: *Advances in neural information processing systems*. 2007, pp. 1209–1216 (cit. on p. 6).

[195] A. Saxena, J. Driemeyer, and A. Y. Ng. “Robotic grasping of novel objects using vision”. In: *The International Journal of Robotics Research* 27.2 (2008), pp. 157–173 (cit. on p. 6).

[196] S. Schaal. “Is imitation learning the route to humanoid robots?” In: *Trends in cognitive sciences* 3.6 (1999), pp. 233–242 (cit. on p. 6).

[197] T. Schmidt, R. Newcombe, and D. Fox. “Self-supervised visual descriptor learning for dense correspondence”. In: *IEEE Robotics and Automation Letters* 2.2 (2016), pp. 420–427 (cit. on p. 52).

[198] J. L. Schönberger, M. Pollefeys, A. Geiger, and T. Sattler. “Semantic visual localization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6896–6906 (cit. on p. 10).

[199] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. “Scene coordinate regression forests for camera relocalization in RGB-D images”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 2930–2937 (cit. on pp. 9, 18, 52–54, 60, 64, 81, 85, 91, 101, 104).

[200] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. “Learning from simulated and unsupervised images through adversarial training”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2107–2116 (cit. on p. 27).

[201] T. Sielhorst, M. Feuerstein, and N. Navab. “Advanced medical displays: A literature review of augmented reality”. In: *Journal of Display Technology* 4.4 (2008), pp. 451–467 (cit. on p. 6).

[202] K. Simonyan and A. Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *CoRR* abs/1409.1556 (2014) (cit. on p. 91).

[203] K. Simonyan and A. Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014) (cit. on pp. 23, 85).

[204] J. Sivic and A. Zisserman. “Video Google: A text retrieval approach to object matching in videos”. In: *null*. IEEE. 2003, p. 1470 (cit. on p. 23).

[205] N. Souly, C. Spampinato, and M. Shah. “Semi supervised semantic segmentation using generative adversarial network”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 5688–5696 (cit. on p. 75).

[206] K. T. Spoehr and S. W. Lehmkuhle. *Visual information processing*. WH Freeman & Company, 1982 (cit. on p. 3).

[207] R. Subbarao and P. Meer. “Nonlinear mean shift over Riemannian manifolds”. In: *International journal of computer vision* 84.1 (2009), p. 1 (cit. on p. 107).

[208] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel. “Implicit 3d orientation learning for 6d object detection from rgb images”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 699–715 (cit. on pp. 6, 26, 27).

[209] M. Sundermeyer, H. Ney, and R. Schlüter. “From feedforward to recurrent LSTM neural networks for language modeling”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23.3 (2015), pp. 517–529 (cit. on p. 70).

[210] Y. Taigman, A. Polyak, and L. Wolf. “Unsupervised cross-domain image generation”. In: *International Conference on Learning Representations* (2017) (cit. on p. 27).

[211] H. Taira, M. Okutomi, T. Sattler, et al. “InLoc: Indoor visual localization with dense matching and view synthesis”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7199–7209 (cit. on pp. 25, 26).

[212] K. Tateno, F. Tombari, I. Laina, and N. Navab. “Cnn-slam: Real-time dense monocular slam with learned depth prediction”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6243–6252 (cit. on p. 8).

[213] S. Thrun et al. “Robotic mapping: A survey”. In: *Exploring artificial intelligence in the new millennium* 1.1-35 (2002), p. 1 (cit. on p. 7).

[214] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. “Domain randomization for transferring deep neural networks from simulation to the real world”. In: *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE. 2017, pp. 23–30 (cit. on p. 27).

[215] C. Toft, E. Stenborg, L. Hammarstrand, et al. “Semantic match consistency for long-term visual localization”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 383–399 (cit. on pp. 10, 52).

[216] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla. “24/7 place recognition by view synthesis”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 1808–1817 (cit. on pp. 24, 25, 43).

[217] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. “Bundle adjustment—a modern synthesis”. In: *International workshop on vision algorithms*. Springer. 1999, pp. 298–372 (cit. on p. 8).

[218] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. “Adversarial discriminative domain adaptation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 7167–7176 (cit. on pp. 27, 75).

[219] A. Valada, N. Radwan, and W. Burgard. “Deep Auxiliary Learning for Visual Localization and Odometry”. In: *International Conference on Robotics and Automation (ICRA 2018)*. IEEE. 2018 (cit. on p. 64).

[220] J. Valentin, M. Nießner, J. Shotton, A. Fitzgibbon, S. Izadi, and P. H. Torr. “Exploiting uncertainty in regression forests for accurate camera relocalization”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 4400–4408 (cit. on pp. 9, 18, 52–54, 60, 91).

[221] V. Veeriah, N. Zhuang, and G.-J. Qi. “Differential recurrent neural networks for action recognition”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 4041–4049 (cit. on p. 70).

[222] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers. “Image-based localization using lstms for structured feature correlation”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 627–637 (cit. on pp. 64, 70).

[223] X. Wang, A. Shrivastava, and A. Gupta. “A-fast-rcnn: Hard positive generation via adversary for object detection”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2017 (cit. on pp. 75, 76).

[224] S. Weik. “Registration of 3-D partial surface models using luminance and depth information”. In: *Proceedings. International Conference on Recent Advances in 3-D Digital Imaging and Modeling (Cat. No. 97TB100134)*. IEEE. 1997, pp. 93–100 (cit. on p. 74).

[225] B. Williams, G. Klein, and I. Reid. “Real-time SLAM relocalisation”. In: *2007 IEEE 11th international conference on computer vision*. IEEE. 2007, pp. 1–8 (cit. on p. 9).

[226] P. Wohlhart and V. Lepetit. “Learning descriptors for object recognition and 3d pose estimation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3109–3118 (cit. on pp. 24, 26, 27, 30, 32, 34, 37).

[227] J. Wu, L. Ma, and X. Hu. “Delving deeper into convolutional neural networks for camera relocalization”. In: *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE. 2017, pp. 5644–5651 (cit. on p. 60).

[228] Z. Wu, S. Song, A. Khosla, et al. “3d shapenets: A deep representation for volumetric shapes”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1912–1920 (cit. on p. 115).

[229] F. Xue, X. Wang, Z. Yan, Q. Wang, J. Wang, and H. Zha. “Local supports global: Deep camera relocalization with sequence enhancement”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 2841–2850 (cit. on p. 70).

[230] A. Yamaji. “Genetic algorithm for fitting a mixed Bingham distribution to 3D orientations: a tool for the statistical and paleostress analyses of fracture orientations”. In: *Island Arc* 25.1 (2016), pp. 72–83 (cit. on p. 95).

[231] J. Yang, H. Li, and Y. Jia. “Go-ICP Solving 3D Registration Efficiently and Globally Optimally,a Globally Optimal Solution to 3D”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.11 (2016), pp. 2241–2254 (cit. on p. 74).

[232] W. Yang, W. Ouyang, X. Wang, J. Ren, H. Li, and X. Wang. “3d human pose estimation in the wild by adversarial learning”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 1. 2018 (cit. on pp. 75, 76, 82).

[233] K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, and D. Yu. “Recurrent neural networks for language understanding.” In: *Interspeech*. 2013, pp. 2524–2528 (cit. on p. 70).

[234] K. M. Yi, E. Trulls, Y. Ono, V. Lepetit, M. Salzmann, and P. Fua. “Learning to Find Good Correspondences”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2018 (cit. on pp. 58, 59).

[235] Z. Yin and J. Shi. “Geonet: Unsupervised learning of dense depth, optical flow and camera pose”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1983–1992 (cit. on p. 10).

[236] S. Zakharov, W. Kehl, B. Planche, A. Hutter, and S. Ilic. “3D Object Instance Recognition and Pose Estimation Using Triplet Loss with Dynamic Margin”. In: *Proceedings of the International Conference on Intelligent Robots and Systems*. 2017 (cit. on pp. 26, 27, 31, 32, 34, 36, 37, 40, 90).

[237] S. Zakharov, B. Planche, Z. Wu, A. Hutter, H. Kosch, and S. Ilic. “Keep it unreal: Bridging the realism gap for 2.5 d recognition with geometry priors only”. In: *2018 International Conference on 3D Vision (3DV)*. IEEE. 2018, pp. 1–11 (cit. on p. 27).

[238] S. Zakharov, I. Shugurov, and S. Ilic. “Dpod: 6d pose object detector and refiner”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 1941–1950 (cit. on p. 6).

[239] A. R. Zamir, T. Wekel, P. Agrawal, C. Wei, J. Malik, and S. Savarese. “Generic 3d representation via pose estimation and matching”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 535–553 (cit. on p. 26).

[240] H. Zen and A. Senior. “Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis”. In: *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE. 2014, pp. 3844–3848 (cit. on p. 91).

[241] W. Zhang and C. Xiao. “PCAN: 3D attention map learning using contextual information for point cloud based retrieval”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 12436–12445 (cit. on p. 25).

[242] Z. Zhang, T. Sattler, and D. Scaramuzza. “Reference Pose Generation for Visual Localization via Learned Features and View Synthesis”. In: *arXiv preprint arXiv:2005.05179* (2020) (cit. on p. 87).

[243] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. “Learning deep features for scene recognition using places database”. In: *Advances in neural information processing systems*. 2014, pp. 487–495 (cit. on p. 23).

[244] L. Zhou, Z. Luo, T. Shen, et al. “KFNet: Learning Temporal Camera Relocalization using Kalman Filtering”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 4919–4928 (cit. on p. 54).

[245] Q.-Y. Zhou, J. Park, and V. Koltun. “Open3D: A Modern Library for 3D Data Processing”. In: *arXiv:1801.09847* (2018) (cit. on p. 102).

[246] Q. Zhou, T. Sattler, M. Pollefeys, and L. Leal-Taixe. “To Learn or Not to Learn: Visual Localization from Essential Matrices”. In: *IEEE International Conference on Rootics and Automation* (2020) (cit. on p. 72).

[247] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. “On the continuity of rotation representations in neural networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5745–5753 (cit. on pp. 41, 114).

[248] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. “Unpaired image-to-image translation using cycle-consistent adversarial networks”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232 (cit. on p. 27).

[249] M. Zolfaghari, Ö. Çiçek, S. M. Ali, F. Mahdisoltani, C. Zhang, and T. Brox. “Learning Representations for Predicting Future Activities”. In: *arXiv preprint arXiv:1905.03578* (2019) (cit. on p. 91).

List of Figures

List of Tables